51CTO.com
技术成就梦想
我 们 只 谈 开 发

开发月刊

Development Monthly

2013年11月
总第032期

VS2013：Web开发人员必须了解的一切

Spring MVC+JQuery+Google Map打造IP位置查找应用

VISUAL STUDIO 2013

由12306.cn谈谈网站性能技术

12306.cn网站挂了，被全国人民骂了。我这两天也在思考这个事，我想以这个事来粗略地和大家讨论一下网站性能的问题。因为仓促，而且完全基于本人有限的经验和了解，所以，如果有什么问题还请大家一起讨论和指正。

51CTO.com
技术成就梦想

开发
月刊

出版方：

北京无忧创想信息技术有限公司

责任编辑：

林师授

封面设计：

苍旭

联系方法：

邮箱：linss@51cto.com

电话：010-68476606-8123

出版日期：2013年11月25日

欢迎来稿，请发送邮件至

linss@51cto.com

主编的话

设备和服务体验有了根本性的变革，才能改变行业对软件开发的处理方式。本期《开发月刊》的专题内容主要详解微软最新发布的Visual Studio 2013全新Editor功能以及ASP.NET及Web工具为重点的总结。Visual Studio 2013有着提高开发人员工作效率的新功能、支持开发Windows 8.1 应用程序、Web开发技术取得新进展、改进对本机代码和托管代码的调试和优化以及扩展ALM功能等。月刊以原创译文为主，把国外最新，最热的技术展现给大家。

同时，开发月刊集合了热门的资源下载，最近热门技术专题以及技术热点资讯。不管朋友们喜不喜欢我们推荐的文章，但是我们会一如既往的把开发最好的内容推荐给网友，与网友共同进步，成长。

祝好！

51CTO开发频道寄语

目录 Contents

2013年11月
总第32期

编程排行 Billboard

专题报道 Special report

原创译文 Original translation

技术热点 Techlogy hot

编者按

TIOBE社区今天发布了2013年9月的编程语言排行榜，Transact-SQL 在本期榜单中取得了历史性突破，挤掉Perl、Ruby编程语言首次闯入排行榜前十。前五名内有了小范围的浮动，上个月排名第五的C#这个月排名第六，他的位置被PHP所取代，也算是意料之中的事，因为PHP最近的势头正劲。

2013年11月编程语言排行榜：日益蓬勃的微软编程语言

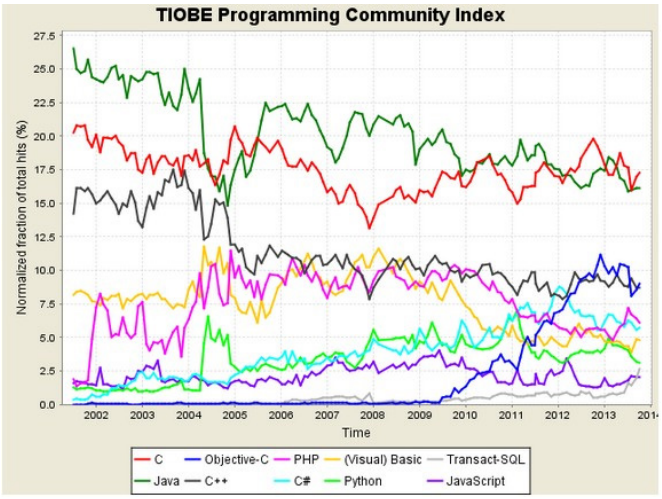
本月编程语言排行榜中，我们看到一件有趣的事。排名前20的编程语言中，四种由微软公司推出的编程语言中，有三种都发展势头迅猛！这四种编程语言是C#、SQL Server 语言、Transact-SQL and Visual Basic.NET。也许这样的现象只是一个巧合。但是，本月Windows Mobile市场占有率同Android和IOS相比有明显的提升。

JavaScript发展依然呈上升趋势。作为时下各种网站必用的编程语言，这样的结果也是意料之中的事。

前20名榜单排行榜

Position Nov 2013	Position Nov 2012	Delta in Position	Programming Language	Ratings Nov 2013	Delta Nov 2012	Status
1	1	=	C	18.155%	-1.07%	A
2	2	=	Java	16.521%	-0.93%	A
3	3	=	Objective-C	9.406%	-0.98%	A
4	4	=	C++	8.369%	-1.33%	A
5	6	↑	C#	6.024%	+0.43%	A
6	5	↓	PHP	5.379%	-0.35%	A
7	7	=	(Visual) Basic	4.396%	-0.64%	A
8	8	=	Python	3.110%	-0.95%	A
9	23	↑↑↑↑↑↑↑↑↑↑	Transact-SQL	2.521%	+2.05%	A
10	11	↑	JavaScript	2.050%	+0.77%	A
11	15	↑↑↑↑	Visual Basic .NET	1.969%	+1.20%	A
12	9	↓↓↓	Perl	1.521%	-0.66%	A
13	10	↓↓↓	Ruby	1.303%	-0.44%	A
14	14	=	Pascal	0.715%	-0.17%	A
15	13	↓↓	Lisp	0.706%	-0.25%	A
16	19	↑↑↑	MATLAB	0.656%	+0.04%	B
17	12	↓↓↓↓↓	Delphi/Object Pascal	0.649%	-0.35%	A-
18	17	↓	PL/SQL	0.605%	-0.03%	A-
19	24	↑↑↑↑↑	COBOL	0.585%	+0.11%	B
20	20	=	Assembly	0.532%	-0.05%	B

前十名编程语言长走势图



后50名的编程语言排行：

(Visual) FoxPro, 4th Dimension/4D, ABC, ActionScript, Algol, Alice, APL, ATLAS, Automator, Awk, bc, BlitzMax, CFML, cg, CL (OS/400), Clean, Clojure, cT, Dart, Eiffel, Emacs Lisp, Euphoria, Forth, GNU Octave, Icon, Inform, Informix-4GL, Io, J, J#, LabVIEW, Max/MSP, Modula-2, Modula-3, Moto, MS-DOS batch, NATURAL, Object Rexx, OCaml, OpenCL, OpenEdge ABL, PILOT, Pure Data, Q, S, S-PLUS, Smalltalk, Standard ML, VHDL, Z shellStandard ML, VBScript, VHDL, X10, Z shell

21-50编程语言排名：

Position	Programming Language	Ratings
21	SAS	0.519%
22	Ada	0.512%
23	F#	0.498%
24	Fortran	0.481%
25	Bash	0.475%
26	ABAP	0.458%
27	C shell	0.445%
28	Ladder Logic	0.435%
29	Logo	0.427%
30	Lua	0.425%
31	R	0.406%
32	Groovy	0.393%
33	D	0.392%
34	Common Lisp	0.387%
35	NXT-G	0.369%
36	Scheme	0.354%
37	Prolog	0.335%
38	Tcl	0.314%
39	PostScript	0.295%
40	VBScript	0.293%
41	Scala	0.292%
42	Erlang	0.272%
43	Scratch	0.269%
44	JavaFX Script	0.261%
45	PL/I	0.251%
46	Haskell	0.250%
47	ML	0.234%
48	RPG (OS/400)	0.224%
49	Go	0.214%
50	JScript.NET	0.212%

微软十大编程语言

1、C++

C++这个词通常被读做“C加加”，而西方的程序员通常读做“C plus plus”，“CPP”。它是一种使用非常广泛的计算机编程语言。

C++是一种支持多重编程范式的通用程序设计语言。它支持过程化程序设计、数据抽象、面向对象程序设计、制作图标等等泛型程序设计等多种程序设计风格。

2、C#

C#是微软公司发布的一种面向对象的、运行于.NET Framework之上的高级程序设计语言。它是由微软工程师Anders Hejlsberg主导开发的。

C#看起来与Java有许多相似之处；它包括了诸如单一继承、接口、与Java几乎同样的语法和编译成中间代码再运行的过程。

但是C#与Java有着明显的不同，它借鉴了Delphi的一个特点,与COM(组件对象模型)是直接集成的，而且它是微软公司.NET windows网络框架的主角。此外，C#还具有安全、稳定、简单的特点。

3、Visual Basic

VB是一款由微软公司开发的包含协助开发环境的事件驱动编程语言，拥有众多的忠实使用者。

它源自于BASIC编程语言，VB拥有图形用户界面和快速应用程序开发系统，可以轻易的使用DAO、RDO、ADO连接数据库，或者轻松的创建ActiveX控件，深受开发人员的喜爱。■

本文为玩，全文阅读请进入以下链接：

http://developer.51cto.com/art/201311/416554.htm

编者按

Groovy、Scala和Clojure提供了许多扩展机制，但继承几乎是Java语言的惟一选择。这一期将介绍类别类、ExpandoMetaClass、隐式转换和协议，借助它们来使用 Java 下一代语言扩展 Java 类。

Visual Studio 2013中的全新Editor功能

在Visual Studio 2013的Editor当中，我们引入了一系列旨在提升生产力并节约时间的新功能。其中一部分属于全新功能，另一部分则来自Productivity Power Tools的高人气扩展。这些功能的加入要归功于用户朋友们通过User Voice建议、论坛帖子以及Connect错误所提出的使用反馈。MVP社区也在扩展项目的选择中向我们伸出援手。

在新版本Editor当中，我们的主要在于保证开发人员尽可能不必脱离当前编辑环境。这篇博文所介绍的功能将在轻松提供必要信息的同时允许大家始终处于当前代码位置。

Peek Definition (Alt + F12)

我们都知道，开发人员在浏览定义时往往需要在代码当中来回游走。在设计Visual Studio 2013的功能时，我们认真考量如何利用元素与手势帮助开发人员在浏览定义的同时又不会失去当前代码位置。Peek Definition就是这样一项功能，允许大家在Editor内部查看定义而无需额外开启新的文档标签。要体验它的实际效果，各位可以右键单击某个符号，在快捷菜单中点选“Peek Definition”命令或者直接按下快捷键组合Alt+F12。

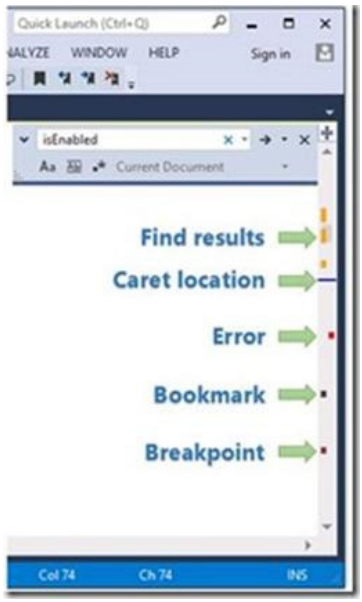
利用Peek Definition能帮助我们节约花费在理解代码库上的时间，因为它允许用户查看相关定义而不必将注意力从当前代码处移开。我们甚至能够在不脱离原始文件的前提下，浏览一系列相关定义内容。当大家在当前Peek视图中调用Peek Definition命令时，系统会引导各位在Peek视图内部查看定义，并在之前的Peek视图中添加返回导航。这些返回导航元素能帮助大家直观了解自己是通过怎样的途径一步步抵达当前定义的。

在设计这项功能时，我们选择以出现在VS 2012中的“Preview Tab”概念为基础，这是因为Peek视图是一种暂时性显示结果、目的在于帮助大家免于开启更多文档。返回导航则是我们添加的另一种引导方案，希望帮助大家在继续面对当前代码的同时直观了解导航堆栈。事实上，如果大家将鼠标悬停在返回导航之上，系统会提示一小段代码，这样各位就能快速了解自己将被引导到哪里。

在Visual Studio 2013预览版中，Peek视图拥有只读属性。经过内部用户的测试，我们发现大家明确希望能够对Peek视图进行编辑——当然，我们立即着手实现这一要求。请朋友们立刻拿起手中的正式版本进行体验吧！

增强滚动条（Enhanced Scrollbar）

增强滚动条（Enhanced Scrollbar）一直是Productivity Power Tools当中最受欢迎的扩展，现在则正式成为Visual Studio 2013中的标准成员。增强滚动条为用户在垂直滚动条上提供可视化线索信息。滚动条上的标记能帮助大家快速查看错误、警告、中断点、书签、查找结果的位置以及其它一些文件中的实用信息。同样，我们希望凭借增强滚动条帮助用户在当前位置了解更多提示结果——而不必将滚动条拖动到对应位置。



地图模式

我们在滚动条中添加了一项“地图模式”（Map Mode），从而带来一些很酷的功能——而且大家可以对该机制进行自定义：

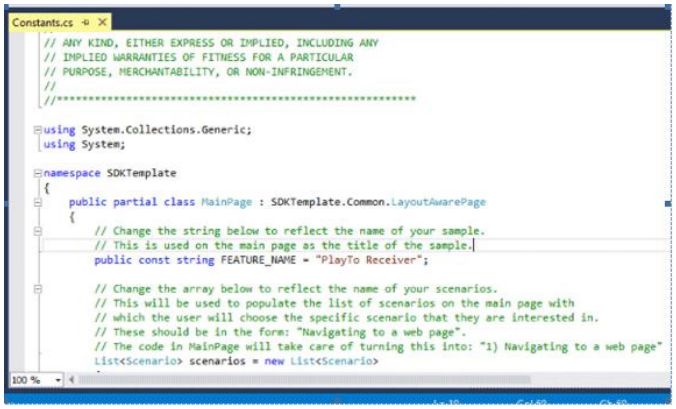
1.长达10000英尺的代码视图 - 对代码进行宏观审视，从而让滚动条成为帮助我们清晰理解代码结构的利器。

2.单击滚动 - 点击滚动条中的任意区域即可直接将大家引导至文件中的对应位置——这是一种快速实现文件内容定位的新机制。

3.预览文件中的特定部分 - 将鼠标悬停在滚动条上的任意位置即可在提示信息中对对应部分的内容进行预览。这项功能在文件审查过程中会带来便捷的使用体验。

导航至(Ctrl+,)

在提高开发生产力的探索过程中，我们尝试摆脱传统的独立工具窗口及模式对话框，转而寻求一种更为流畅的直接体验，从而帮助大家将注意力集中在编码工作上而非管理Visual Studio。我们对使用数据进行了分析并决定对导航至（Navigate To）机制进行更新，从而让这套目前仍被广泛使用的模式对话框更上一层楼。



通过新的导航至功能，大家可以输入任意一部分符号内容并利用智能语义搜索查找其定义。大家还可以输入一部分文件名以快速切换到对应位置——

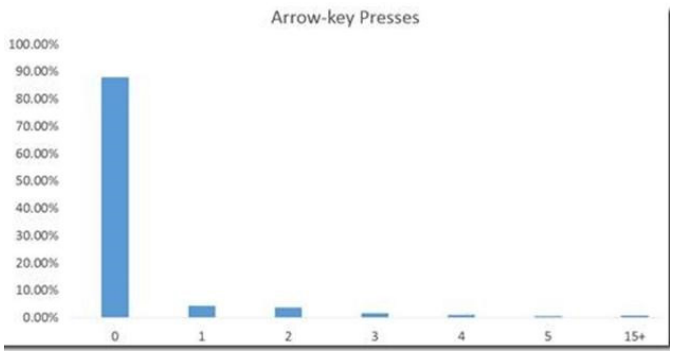
无论该文件之前是否已经被打开。

Visual Studio 2013中的导航至功能支持早期版本的所有功能，但转而以一种更流畅、非模式化且空间利用率更高的方式呈现。我们把新的搜索窗口放置在编辑器内查找区周边的右上位置。这种布局方式让我们能够在显示预览标签的同时尽可能增加导航至结果的显示数量，而且不会遮挡预览代码视图。

选择的结果将自动显示在预览标签当中。这能帮助用户确保被选中的结果正是自己搜索的对象，从而在将其提交至新视图之前更好地做出判断。为了确保大家能够更轻松地返回原先位置，我们只需按下Esc键即可取消跳转结果——这就避免了搜索结果错误可能带来的定位困扰。

在设计这些功能时，我们不只希望改进原始运行时性能，同时也期待改善工作流。新的导航至功能针对键盘操作使用情况进行了优化，即参考到Solution Explorer的浏览重点。在功能开发的过程中，我们检测并分析了内部用户的实际使用状况，从而检验新的功能设计是否具有实际效率。下面请大家一同了解我们所参考的统计数据。

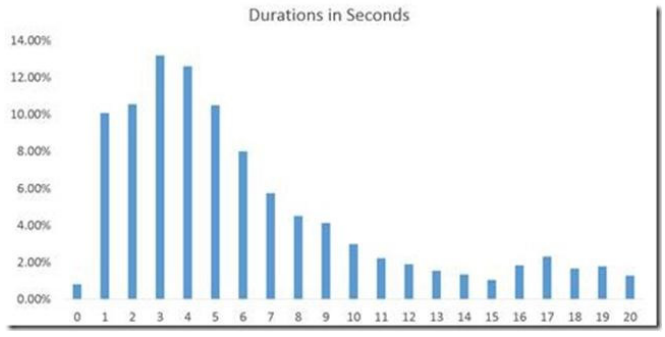
我们统计了用户使用键盘上“下箭头”按键的次数，并以此结果作为提示列表准确性的评判依据：



数据显示在大多数情况下，用户都只需直接按下回车键即可——这意味着搜索查询实现了准确定位。也有些用户在浏览结果时按下15次以上“下箭头”按键，这其实与准确性无关——只是他们查找的结果数量比较多而已。

为了决定在用户点击结果列表之外的区域时是否继续开启导航至功能，我们统计了用户保持其开启的次数。经过调查，我们发现从对话框开启那一刻到其关闭，平均持续时间为六秒钟：数据显示在大多数情况下，用户都只需直接按下回车键即可——这意味着搜索查询实现了准确定位。也有些用户在浏览结果时按下15次以上“下箭头”按键，这其实与准确性无关——只是他们查找的结果数量比较多而已。

为了决定在用户点击结果列表之外的区域时是否继续开启导航至功能，我们统计了用户保持其开启的次数。经过调查，我们发现从对话框开启那一刻到其关闭，平均持续时间为六秒钟：



这反映了两种最常见的操作情况：用户导航到新位置并开始浏览，或者取消了该操作。这样的结果让我们决定在用户点击其它区域后关闭导航至对话框。我们会继续关注使用数据以确保这种设计符合大多数用户的直观感受。

自动补全括号

自动补全括号功能，顾名思义，会自动为我们在编辑器中输入的代码补齐右侧括号、引号、大括号等。这也是来自Productivity Power Tools的一项

高人气功能，现在我们将其正式引入Visual Studio 2013。

字符串的字面表达并利用“*/”作为C类注释的结束标记，同时在类类型中插入分号。

上/下行移动 (Alt+上箭头/下箭头)

上/下行移动功能允许大家快速向上向下移动一行或者多行，具体操作方式为Alt+上箭头以及Alt+下箭头。这是Productivity Power Tools中的另一项人气扩展，此次也加入了Visual Studio 2013。

```
if (shouldPrint)
{

    string hello = "";
    hello += "Hello world";

}
```

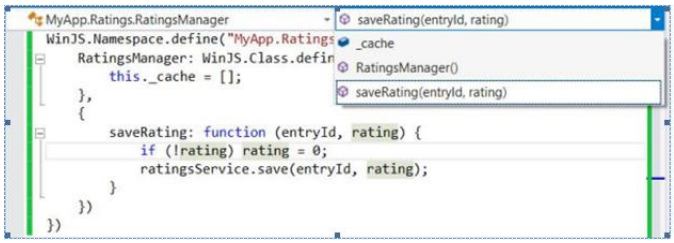
针对Visual C++的新IDE功能

我们还针对Visual C++添加了多项新功能。在未来几周中，Visual C++技术团队将在其博客上分享更多信息，请大家保持关注。不过在今天的文章中，我们要强调的是C++代码格式。

C++编辑器现在能在输入的同时自动调整格式，这一功能也作用于被直接粘贴到C++文件中的代码。我们发现对于C++来说，目前还没有一种被广泛接受的编码风格，因此我们在设置中添加了灵活性选项，允许大家对环境进行自定义以匹配自己的开发风格。我们期待您给出的反馈意见，这样才能对设置做出进一步调整。

针对JavaScript的新IDE功能

在JavaScript当中，我们添加了标识符高亮功能——现在当大家选择某个标识符（例如变量名称或者函数调用）时，对它的引用关系将被以高亮方式显示在当前源文件中。我们还设置了一个新的导航栏，用于在编辑器窗口上方显示相关内容，这样大家就能更轻松地在JavaScript源文件当中在主函数与对象之间来回切换。■



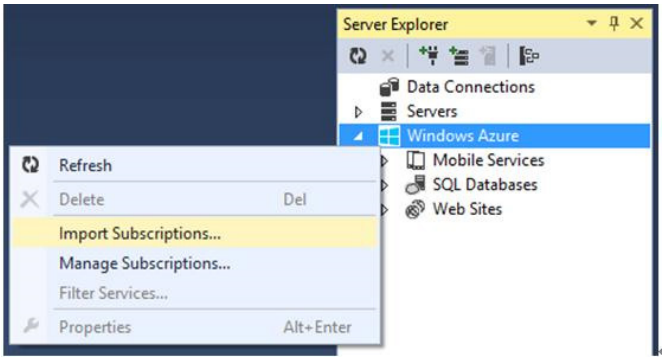
Visual Studio2013创建、公布监控Windows Azure网站

■ Java 下一代语言（Groovy、Scala 和 Clojure）的共同点多于不同点，主要集中于很多功能和方便性上的共同点。本期文章探究它们各自如何克服 Java™ 语言中长期存在的一个缺点 — 无法重载操作符。还要讨论关联性和优先级等相关概念。

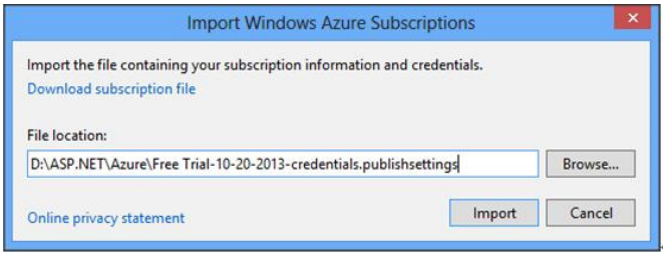
随着Visual Studio 2013的发布，现在我们可以在Visual Studio内部实现Windows Azure网站的创建、发布与监控工作，而且完全无需打开Windows Azure门户。虽然我们在Visual Studio 2012中就已经能够直接实现Azure网站的发布，但在新版本的支持下对Azure网站的创建和监控工作变得更为便捷。我们需要进行的准备工作只有一项——订阅Windows Azure。如果大家还没有订阅Azure服务，可以点击此处进行免费试用。本文将分为三个主要部分，分别为创建Azure网站、发布到Azure网站以及监控Azure网站——当然，这一切都将在Visual Studio 2013当中进行。

创建Azure网站

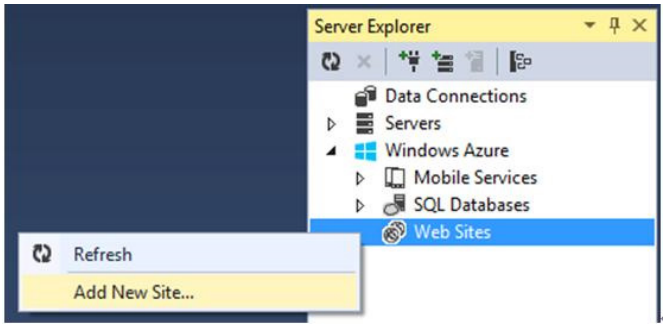
要创建全新Azure网站，我们首先需要打开Server Explorer、右击Windows Azure选项并点选Import Subscriptions…



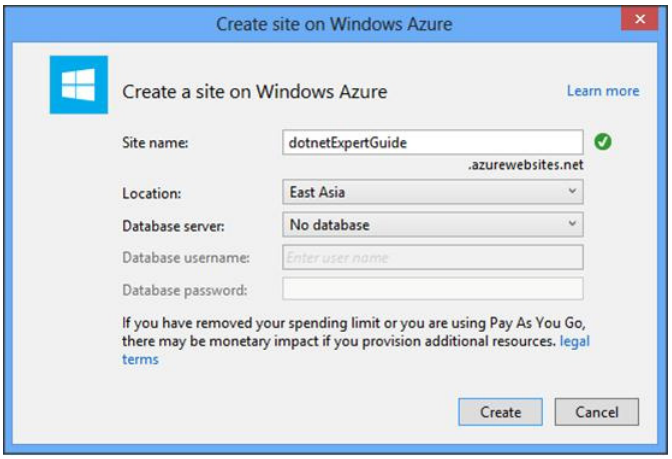
在Import Subscription对话框中点击Download subscription file。系统会打开Windows Azure门户并下载配置文件。游览下载的文件并单击Import。



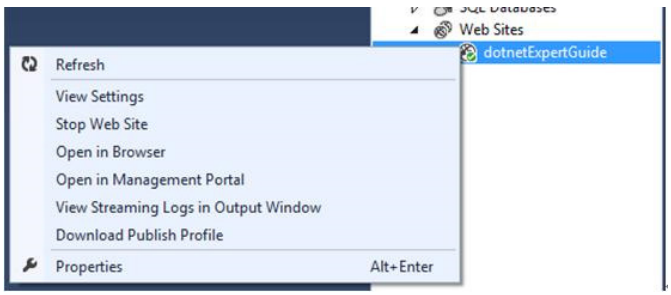
点击Import按钮后，系统会导入对应订阅信息。我们可以在这里管理移动服务、SQL数据库以及Azure上的网站。要创建新的Azure网站，右键点击Web Sites选项并选择Add New Site…



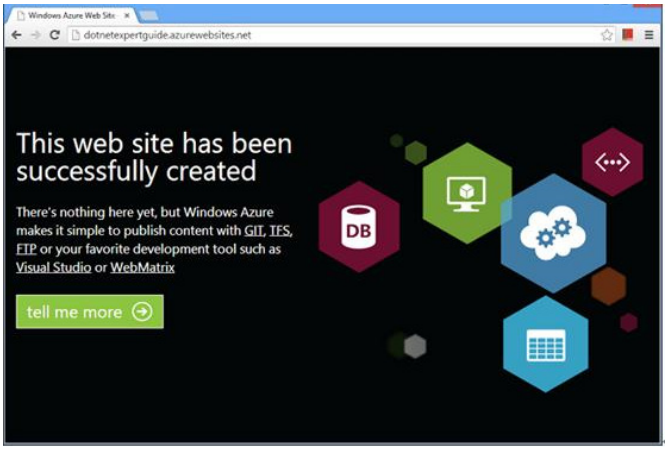
在Windows Azure对话框的Create Site中提交必要的网站信息并单击Create按钮。



在Azure上创建并配置网站的过程将耗时几分钟。现在在Server Explorer中右击刚刚创建的网站并点选Open in Browser。



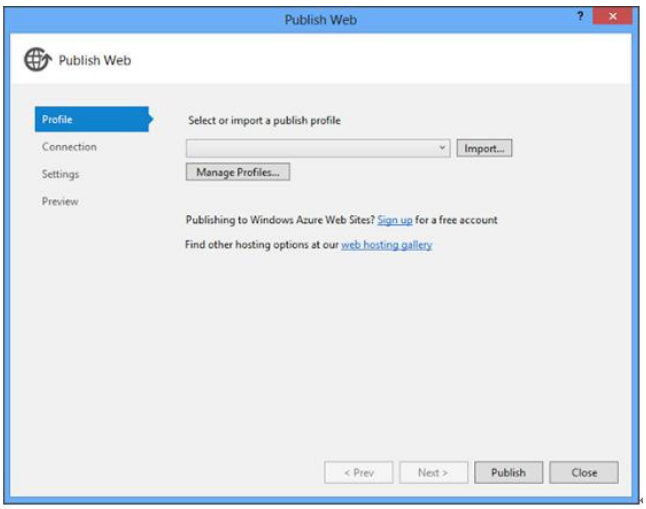
系统会打开浏览器，我们看到如下图所示的新网站默认主页。



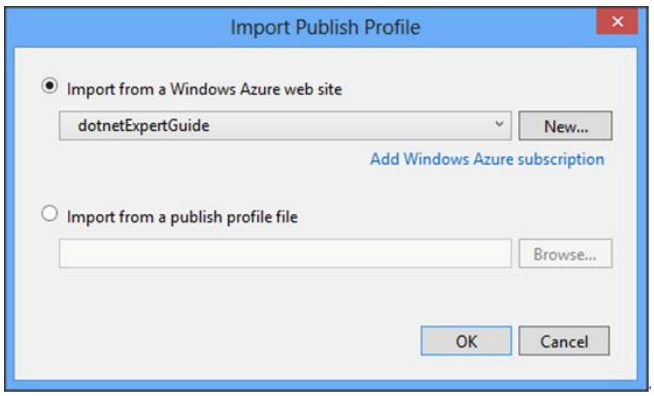
好了，只经过几次点击操作，我们的（空白）Azure网站就创建完成了。下面我们来学习如何直接将ASP.NET应用程序发布到Azure服务器端。

发布到Azure网站

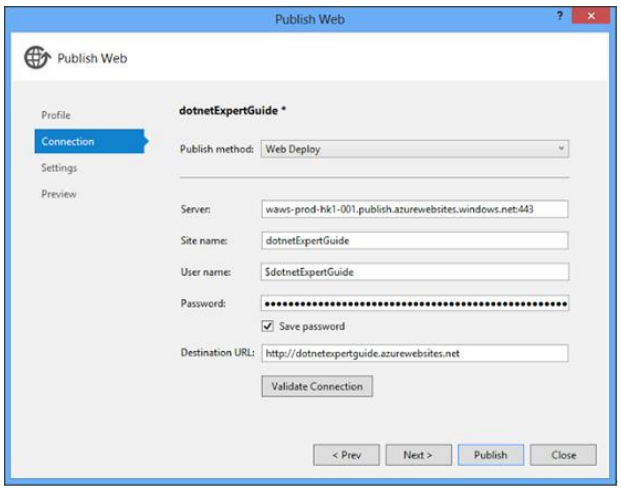
要在Visual Studio 2013当中创建并配置ASP.NET项目，推荐大家点击此处阅读我的另一篇博文。应用程序准备就绪之后，右击ASP.NET应用并点选Publish…



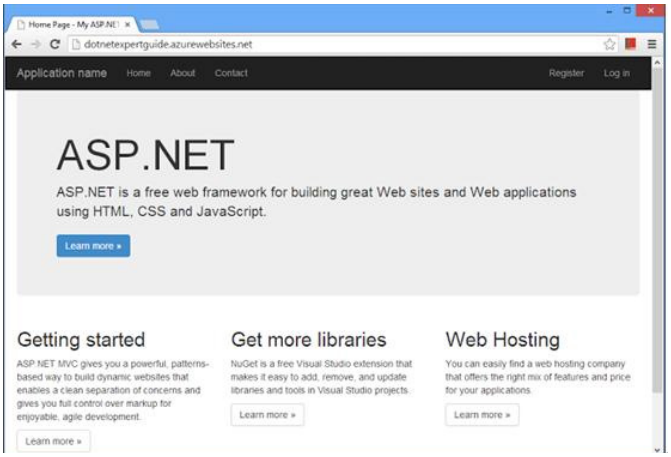
系统会启动Publish向导。点击Import按钮并在下拉菜单中选择要发布的Windows Azure网站，点击OK。



系统会为发布向导中Web部署收集必要信息。



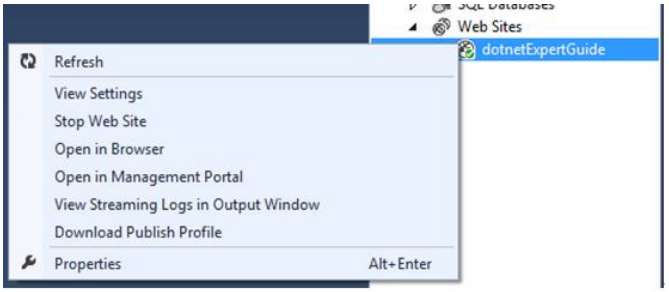
点击Publish 按钮开始向Azure服务器的发布过程。整个过程大约耗时几分钟。发布完成之后，我们可以在浏览器中打开网站以检验发布是否成功。



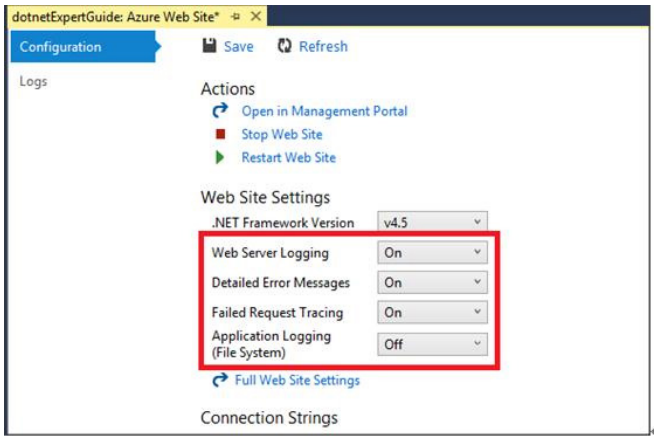
现在我们已经完成了将网站向Azure服务器发布的工作。在下一节中，我们将探讨如何通过Visual Studio对Azure网站实施监控。

监控Azure网站

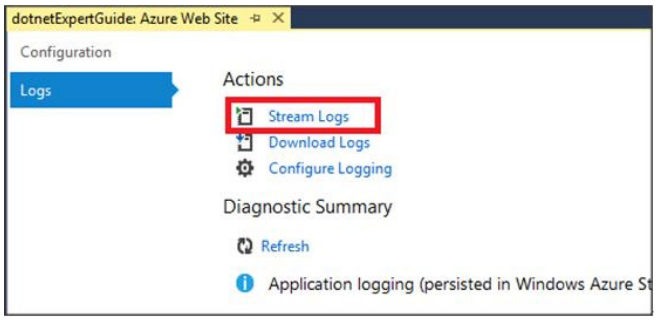
要监控Azure网站，需要右键点击该网站并选择View Settings。



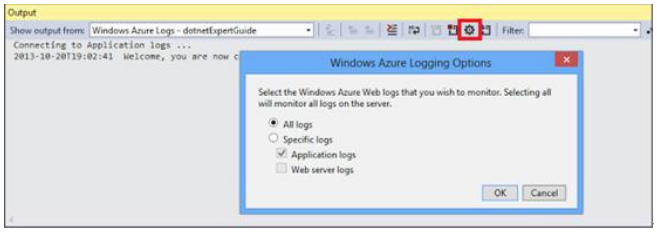
系统会打开网站设置面板，在这里我们可以管理并配置Azure网站，包括启动/停止/重启网站、配置框架版本以及启用日志记录等。我们还可以从这里直接打开Windows Azure门户。



为了监控网站，大家必须关注上图中标出的日志记录选项。在启用日志记录功能后，进入设置面板中的Logs标签。



在这里我们可以通过右键点击Download Logs下载日志信息。要对网站进行实时监控，则需要点选Stream Logs。



系统将打开Output窗口。如上图中的标注区域所示，点击输出窗口中的设置按钮并选择All Logs，这样输出窗口就会开始捕捉并显示Azure网站的实时日志内容。要查看实时日志，大家需要发布Azure网站、如上所述启动日志捕捉功能并在输出窗口中访问最新创建的两到三页内容。

好了，到这里我们已经可以通过输出窗口以实时方式捕捉并显示网站日志了。■

用Visual Studio 2013开始MVC5之旅

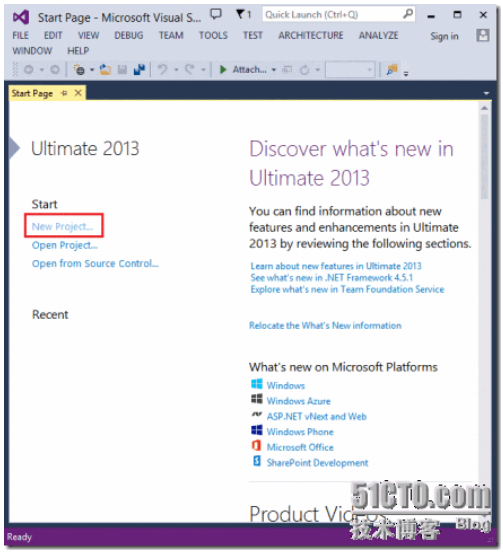
Visual Studio是一个IDE集成开发环境。就像您使用Microsoft Word来编写文档，你可以使用集成开发环境(IDE)来创建一个应用程序。在Visual Studio中的一个顶部工具栏中显示了各种不同的选项来供您使用。

本教程将使用Visual Studio 2013手把手教你构建一个入门的ASP.NET MVC5 Web应用程序。本教程配套的C#源码工程可通过如下网址下载：C#版本源码链接。同时，请查阅 Building the Chapter Downloads 来完成编译源码和配置数据库。在本教程中的源码工程，您可在Visual Studio中运行MVC 5应用程序。您也可以使Web应用程序部署到一个托管服务提供商上。微软提供免费的网络托管多达10个网站，free Windows Azure trial account。本教程由Scott Guthrie (twitter @scottgu), Scott Hanselman (twitter: @shanselman), and Rick Anderson (@RickAndMSFT)共同写作完成，由葡萄城控件团队翻译(新浪微博 @葡萄城控件)。

入门

运行 Visual Studio Express 2013 for Web 或 Visual Studio 2013开始这个实例。

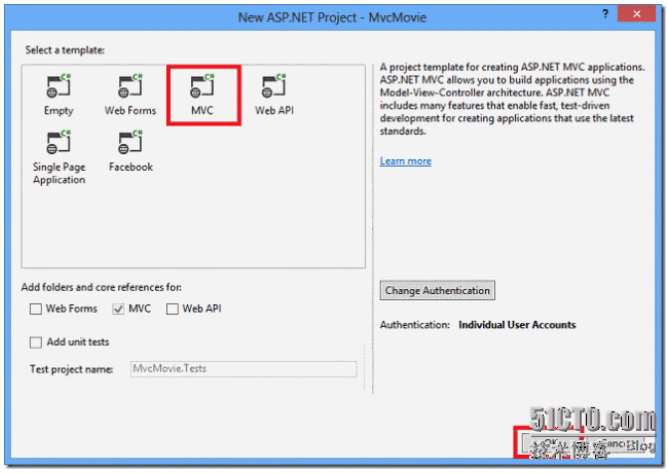
Visual Studio是一个IDE集成开发环境。就像您使用Microsoft Word来编写文档，你可以使用集成开发环境(IDE)来创建一个应用程序。在Visual Studio中的一个顶部工具栏中显示了各种不同的选项来供您使用。在IDE中还有一个菜单，提供了另一种方式来执行任务。(例如，您可以不从“开始”页面中，选择“新建项目”，您可以使用该菜单，然后选择“文件“>“新建项目“)



创建您的第一个MVC 5应用程序

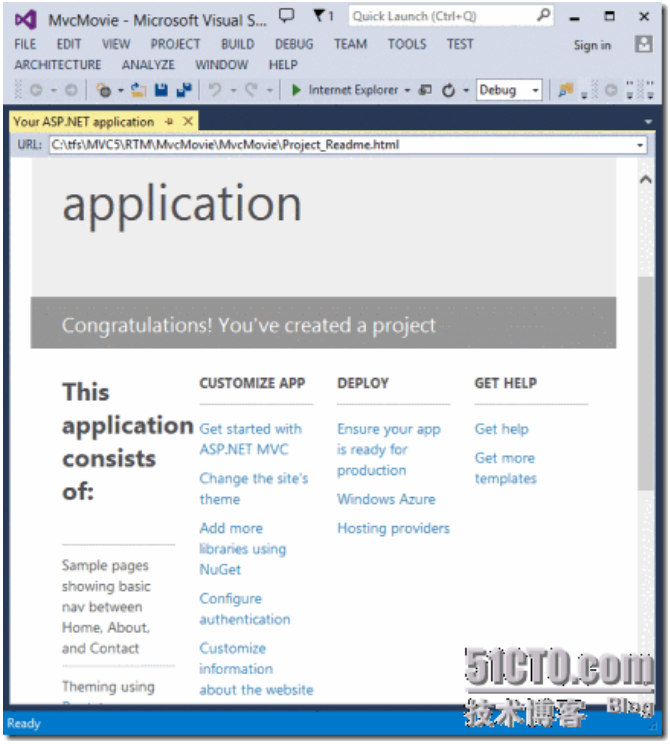
点击新建工程，在左侧选择Visual C#，接着选择Web，然后选择ASP.NET Web Application. 命名您的工程为”MvcMovie”，然后单击确定。

在 New ASP.NET Project 对话框, 选择 MVC 模板，然后单击确定。



Visual Studio 刚刚创建的 ASP.NET MVC 项

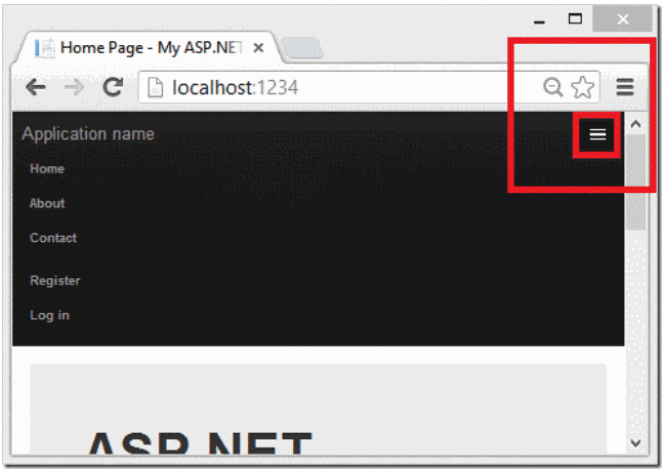
目使用了默认的模板，所以在当前的工程中您不需要做任何事情!这是一个简单的”Hello World !”工程，并且这也是您开始“MvcMovie”工程的好地方。



按下键盘快捷键 F5开始启动调试。 F5使得Visual Studio启动IIS Express并运行Web应用程序。然后Visual Studio会启动浏览器并打开应用程序的主页面。请注意，在浏览器的地址栏中会显示localhost:port#而不是像example.com 这样的地址。这是因为 localhost总是会被解析为您自己的本地计算机，在这种情况下，这正是您你刚刚建立的应用程序。当Visual Studio运行一个Web工程时，会使用一个随机端口的Web服务。在下面的图片中，端口号是1234。当您运行该应用程序时，您可能会看到一个不同的端口号。

在默认模板页面的右边，为您提供了“主页(Home)”，“关于(About)”和“联系(Contact)”页面。下面的截图没有看到“主页(Home)”，“关于(About)”和“联系(Contact)”连接。这取决于你浏览器窗体的大小，你可通过点击右上角导航图标看到这些链接。

同时，默认模板创建的ASP.Net MVC应用程序还提供了注册和登录功能。 接下来的一步是修改此默认应用程序，并了解一些关于ASP.NET MVC的知识。关闭浏览器，让我们修改一些源代码吧。■



VS2013：Web开发人员必须了解的一切

【51CTO独家译文】Visual Studio 2013已经发布，我们准备了下面这份以ASP.NET及Web工具为重点的Visual Studio 2013版本总结：

· Visual Studio 2013现在已经可通过**Visual Studio网站**进行下载，MSDN订阅用户则可**点击此处**下载。

· Visual Studio 2013需要与Visual Studio 2012配合安装，且允许用户在不同Visual Studio版本之间随意切换——因此大家完全不必存在顾虑。

· Visual Studio 2013带来新的ASP.NET版本，其中包括ASP.NET MVC 5、ASP.NET Web API 2、Razor 3、Entity Framework 6以及SignalR 2.0。

· ASP.NET新版本的关注重点在于“One ASP.NET”，因此其核心功能及网络工作能够在不同平台之上实现同样的效果(例如将ASP.NET控制器添加到Web Forms应用当中)。

· 新增核心功能包括基于Bootstrap的新模板、新的支架系统以及新的身份识别系统。

· Visual Studio 2013是一款极为出色的编辑器，能够打理各种Web文件，包括HTML、CSS、JavaScript、Markdown、LESS、Coffeescript、Handlebars、Angular、Ember、Knockdown等。

热门链接：

· ASP.NET网站标准新版本区中的Visual Studio 2013相关内容：http://www.asp.net/vnext

· Visual Studio 2013版本中的ASP.NET与Web工具说明

· Scott Hanselman与Mads Kristensen带来的Visual Studio网络编辑器功能简要介绍(视频)

· .NET Web开发与工具官方博客中的Visual Studio 2013中ASP.NET与Web工具新版本博文

.NET Web开发与工具博客提供很多关于新版本的优秀博客

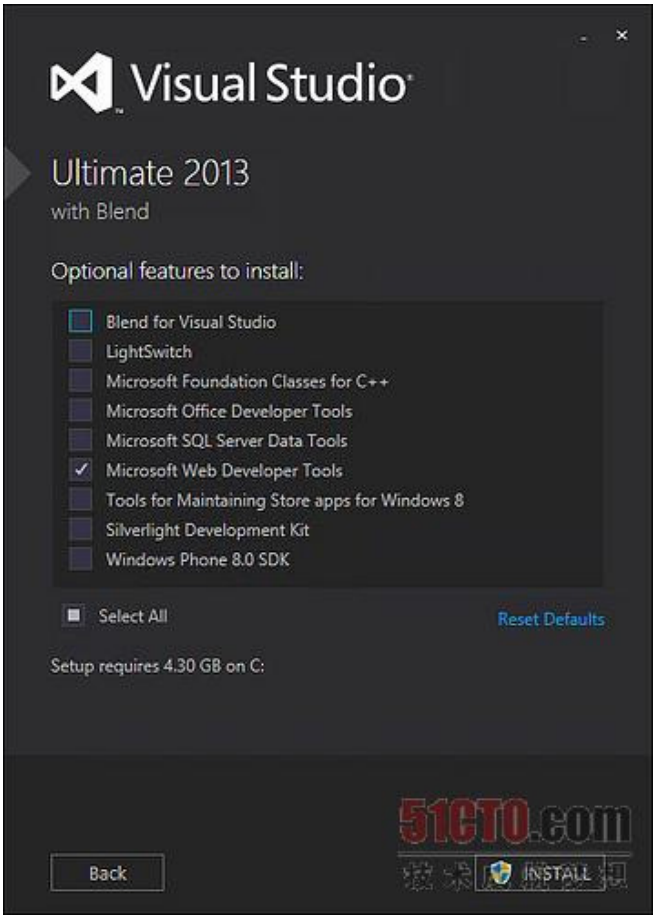
· Scott Guthrie的博客：Visual Studio 2013版本、ASP.NET与Enitity Framwork的新改进

· Scott Hanselman在ASP.NET网站上发布了一系列功能概述视频。它们做得非常好，能够带来理想的指导作用。大家可以通过以下地址进行观看：http://www.asp.net/visual-studio/overview/2013

Web开发指导：Visual Studio 2013的下载与安装

我发现Visual Studio 2013的安装过程相当快。根据Brian Harry发布的博文来看，直接安装并覆盖Visual Studio预发布版也是完全可行的。

如果大家打算利用Visual Studio 2013单纯进行Web开发，那么完全可以通过勾选“Web Developer Tools”选项来进一步缩短安装流程。



当然，我肯定还得介绍介绍其它能帮助用户完成任务的功能，例如针对Windows 8系统的Store应用以及Windows Phone 8.0 SDK——不过它们会下载并安装大量其它组件(举例来说，Windows Phone SDK会设置Hyper-V并下载几GB虚拟机数据)。因此，如果大家的本意仅仅是为了进行Web开发，那么只选择Web Developer Tools即可——其它组件可以稍后再进行安装。

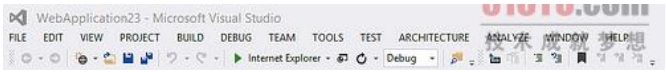
如果各位手头的互联网连接比较给力，我建议大家直接使用网络安装工具而不必下载整个ISO。ISO文件当中包含全部功能，但网络安装工具则只会下载被选中的对应组件。

Visual Studio 2013开发设置与色彩主题

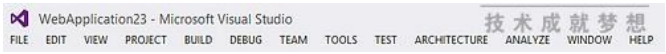
当大家首次启动Visual Studio时，它会提示我们选择一些默认设置。具体如何选择完全取决于大家的喜好——只要符合具体开发习惯即可——各位也可以在稍后的使用中对其进行修改。

正如刚刚所说，具体设置完全取决于个人喜好。我建议大家在Web开发与Web开发(纯代码)之间进行选择。二者的惟一实质性区别在于纯代码模式会隐藏工具栏，大家可以利用Tools/Import与Export Settings/Reset在两种模式之间切换。

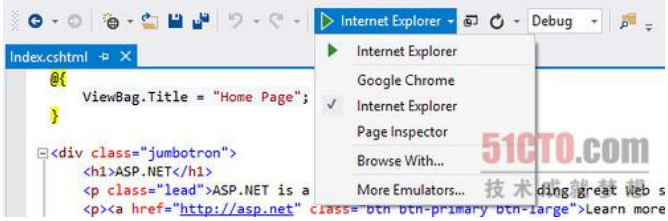
Web开发设置



Web开发(纯代码)设置



通常情况下，我更倾向于使用Web开发(纯代码)模式，这是因为我希望能把注意力集中在代码身上——当然，标准工具栏能够更轻松地切换为默认网络浏览器视图。我们稍后再进一步探讨。



色彩主题

每个人都可以任意选择自己喜欢的色彩主题，我个人会根据自己的心情选择Light或者Dark色调——此外，我个人很喜欢在低对比度窗口显示方

我个人会根据自己的心情选择Light或者Dark色调——此外，我个人很喜欢在低对比度窗口显示方案，这能让自己的注意力更集中在代码本身而非标签或工具栏身上。我知道有些朋友对这种设定极为不满，希望蓝色主题能马上回归。我倒是对蓝色主题没啥好感——它让我感觉自己仍然置身于远古时代的Visual Studio版本当中……好了，我一秒钟也不打算再多想。

因此，目前的情况是：如果大家安装Visual Studio旗舰版，那么主题默认为Blue，其它版本则默认采用Light。如果各位坚持使用Blue方案，我也不会妄加品评。主题变更非常简单，按照Tools/Options/Environment/General的顺序一路点下去即可;或者选择更聪明的办法：按下Ctrl+Q组合键快速激活，然后输入主题名称并按回车确认。

登录

在第一次使用时，Visual Studio 2013会提醒我们登录。其实不登录也，大家可以点击对话框下方的“暂时不登陆，以后再说”链接。不过我个人建议大家登录。这套登录机制并不会与授权许可挂钩，也不会追踪大家的使用情况并推送功能组件购买推荐。可以说登录只会带来好处，例如在不同计算机之间同步Visual Studio设置等。总之，可以不登录，但登录之后回报多多。

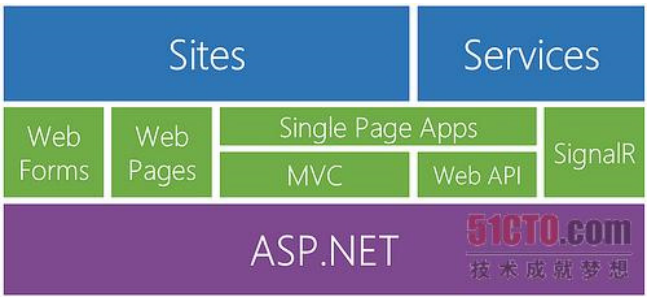
ASP.NET全新内容概述

ASP.NET中的新鲜好东东也不少。我会在接下来的文章中介绍一部分，但更多内容推荐大家访问ASP.NET官方网站。

One ASP.NET

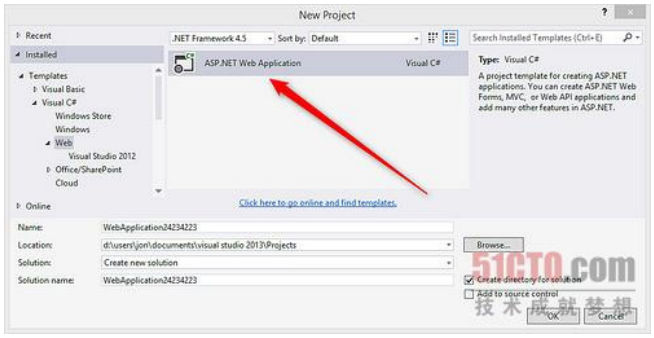
关于这个话题我们已经进行过多次讨论，最终的结论是——有选择是好的，但选择本身也属于一种负担。哪大家着手创建一个全新ASP.NET项目，为什么还要做出一个个艰难的选择(且会造成长期影响)来决定应用程序该如何运作?我们为什么就不能先使用ASP.NET Web Forms，并在未来需要的时候添加ASP.NET MVC——这真有那么难吗?毕竟这手心手背都是ASP.NET，对吧?

在理想状态下，我们只需在项目起步时决定采用ASP.NET作为网站及服务的创建基础，并在其后的实际开发过程中根据需要添加其它工具(即下图中的绿色方块)。

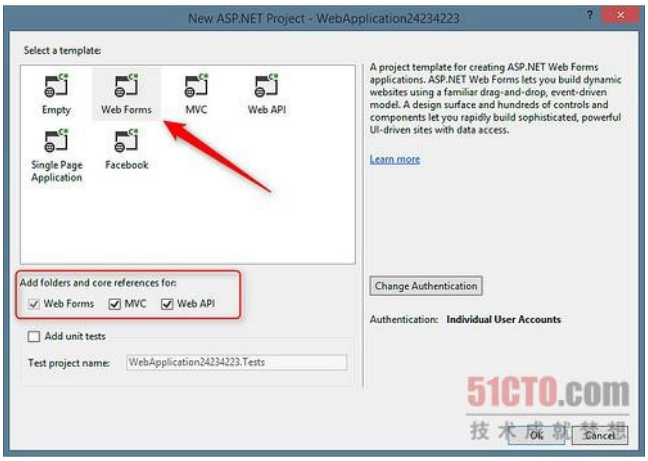


好的，现在美梦终于成真。

当大家创建一个全新ASP.NET应用程序时，只需点击对应选项即可。



接下来，大家可以从现有模板当中选择一种作为起步……不过模板之间的区别可不小。这并不是那种会让大家左右为难、不知如何取舍的模板，而只是一些首发选项。最重要的是，大家可以对其进行混合及匹配。举例来说，我们可以在选择使用Web Forms模板中大部分设定的同时，另行借用MVC及Web API文件夹与核心。



大家过去可能也尝试过混合及匹配方案——没错，虽然能够生效，但使用过程并不愉快。原先的ASP.NET项目文件当中包含各种特殊的项目类型GUID，因此如果大家希望手动编辑csproj文件、则只能在Web Forms项目当中获得控制器支持。同一堆栈中的功能无法作用于其它堆栈，因此选择项目模板让人头痛不已。现在情况不同了，让我们尽情欢呼吧!

我曾在上周的演示环节中拿出一套开发demo——一个由Web Forms加MVC再加Web API创建的网站，内置模块化支架式MVC以及由EF Code First实现的Web API控制器;数据被添加到MVC视图当中，在Web API内进行查看，而后再向Web Forms Default.aspx页面添加一套GridView并与Model绑定。整个过程大约耗时五分钟。当然，这只是个简单的实例，但确实能够在跨ASP.NET家族间进行数据共享方面发挥出色作用。

验证

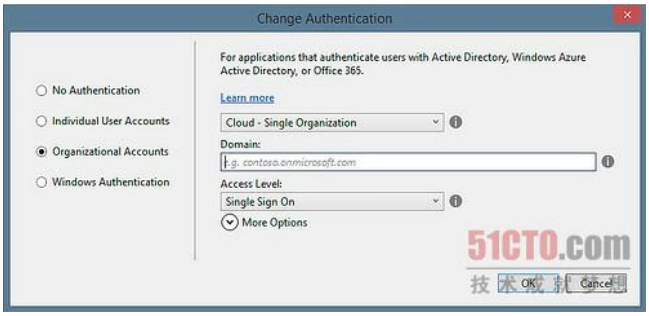
过去，验证机制被内置在模板当中。举例来说，大家需要利用ASP.NET MVC 4 Intranet Project模板来创建新的ASP.NET MVC 4应用程序，其中将预置Windows验证机制。因为关于验证的一切都被内置在每套模板当中，因此它们之间处于不同堆栈范畴，大家无法对其进行重复使用。有鉴于此，关于验证的选项并不多，因为它们涉及一大堆项目模板且需要做出大量改动。

现在，新项目对话框中包含一套通用的验证机制。当大家点击“Change Authentication”(变更验证)按钮时，即可获得通用选项——其起效机制完全相同，不再需要考虑具体模板或者用户的个人设置。这些选项都以ASP.NET框架为基础，且适用于全部托管环境(包括IIS、IIS Express或者OWIN)。

默认情况下使用的是个人用户账户:

这是一套标准的“创建本地账户、使用用户名/密码或者OAuth”机制;不过这一切都以新型验证系统为基础。我们稍后再进一步讨论。

这里只有组织账户这一项需要进行额外配置，包括Active Directory、Windows Azure Active Directory或者Office 365的验证配置。



身份认证

Visual Studio 2013采用新的身份认证系统。在继承了原先ASP.NET Membership以及Simple Identity系统的优势之外，新版本还采纳

了大量反馈意见从而为开发人员带来理想的支持力度，包括单元测试以及扩展能力。

我曾经就ASP.NET的身份认证机制撰写了几篇长博文，在今后的文章中我还将旧事重提。概括来说，我认为我们终于获得了一套恰到好处的身份认证系统，其中最出色的功能包括：

·默认项目简单、合理且效果突出——大家可以按照File/New/Run/Register/Login的顺序设置自己的认证机制，而且肯定能正常起效。

·它支持标准的用户名/密码模式，同时也可采用外部验证机制(例如OAuth等)。

·定制工作容易实现且无需对全盘进行重新实施。它采用内置插接式组件，无需涉及大型独立系统。

·它采用IUser及IRole等作为内置接口，从而实现单元测试、关联性注入等功能。

·大家可以轻松添加用户配置数据(例如URL、Twitter引用、出生日期)。大家只需将属性添加到ApplicationUser模块当中，它们就会自动得到保存。

·对身份认证数据进行持续全面控制。在默认情况下，都由Entity Framework Code First负责打理，但它也能够直接支持由小(修改架构)到大(使用其它ORM或者在文档数据库、云环境、XML、桌面后台EXIF数据或者其它机制中保存数据)的各类变更。

·配置工作通过OWIN实现，稍后Katana也将加入进来。利用OWIN打造意味着它具备便携性。

大家可以在ASP.NET网站上参阅更多与验证及身份认证相关的章节(稍后还将发布更多消息)。

项目模板基于新型Bootstrap

前面提到的各种新型项目模板由Bootstrap 3创建。Bootstrap(即原先的Twitter Bootstrap)是一套前端框架，它带来的优势包括：

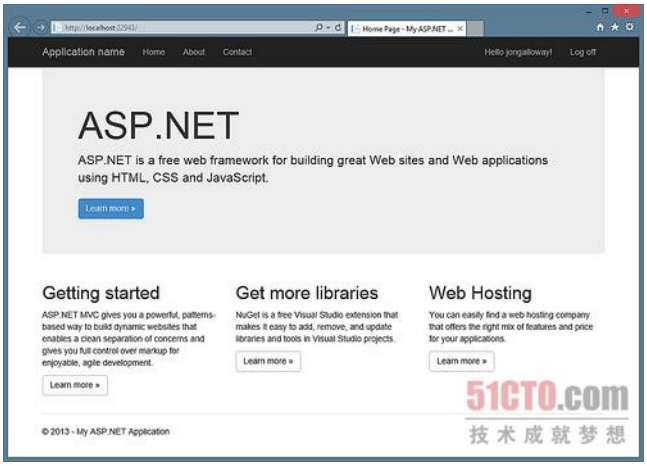
·采用响应式设计，因此项目能够自动利用CSS媒体查询与设备宽度相匹配。举例来说，菜单会在桌面浏览器中以全尺寸显示;但在移动设备的小型屏幕上，大家会自动获得符合移动使用习惯的菜单方案。

·内置Bootstrap风格将使大家的标准页面元素(包括标题、页脚、按钮、表单输入以及列表等)外观更漂亮、更具现代感。

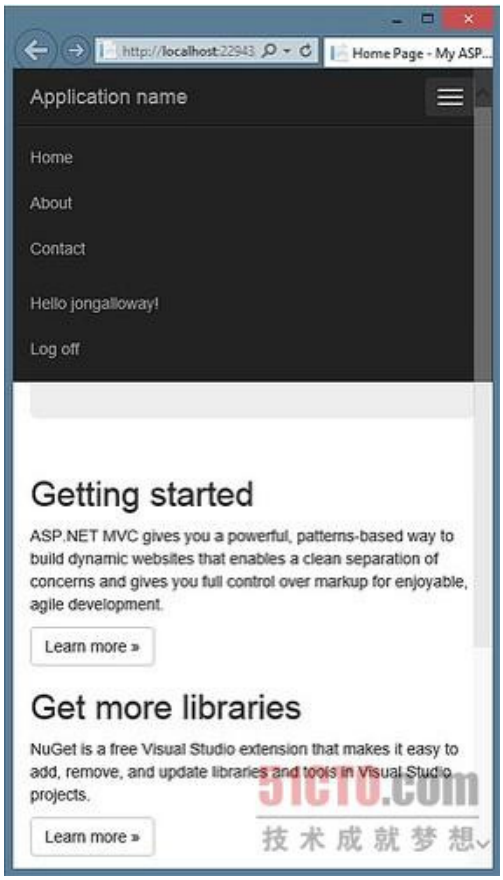
·Bootstrap支持主题设置，因此大家可以在下拉菜单的新Bootstrap主题选项中重新装扮自己的网站。由于Bootstrap在Web开发行业中的人气很高，因此各类模板数量庞大、品种丰富——付费版本与免费版本皆有，任君随意选择。

·Bootstrap还包含大量极具实用性的其它内容：组件(例如进度条及标记)、glyphicons、用于工具提示的jQuery插件、下拉菜单及carousel等

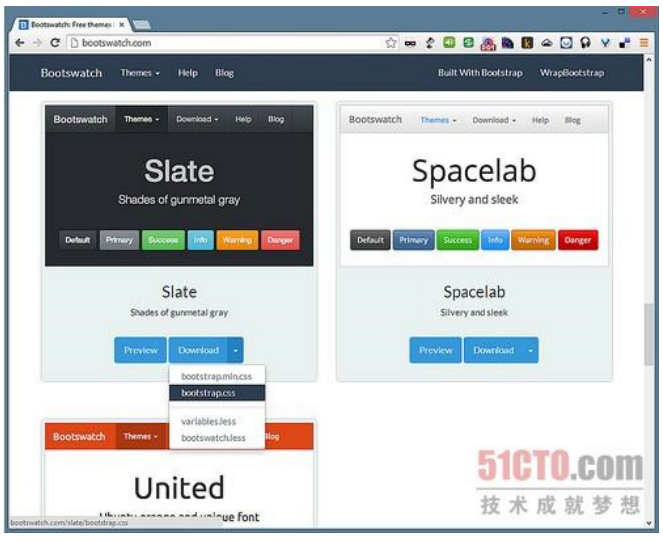
下面我们来看响应式部分是如何工作的。在页面处于全屏模式下时，菜单与标题会针对宽屏显示进行优化：



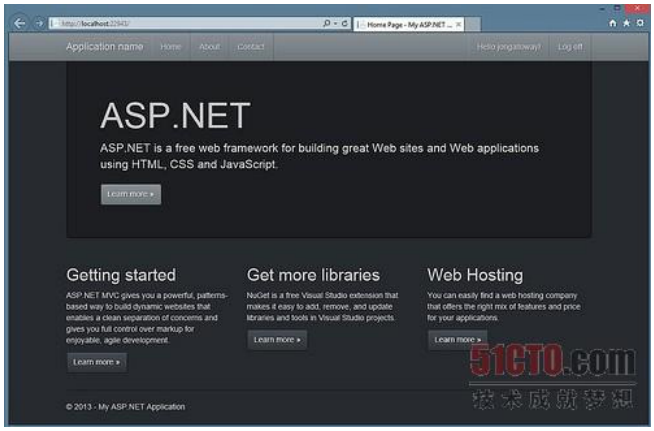
在缩小页面尺寸之后(完全取决于页面宽度，与useragent检测无关)，菜单会调整为更适合在移动设备上观看的纵向模式：



举例来说，我从bootswatch.com网站上找到一套新的免费主题。对于简单主题来说，大家只需要下载其bootstrap.css文件并将其覆盖项目中的/content/bootstrap.css文件即可。

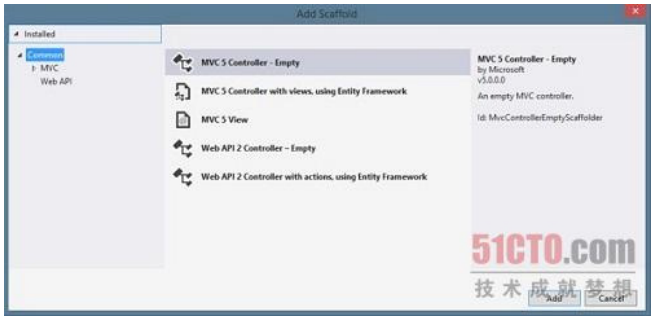


现在只需刷新当前页面，新主题就会自动生效：



支架

支架系统的最大变化在于可以在整套ASP.NET体系下发挥作用。大家可以创建一个新的空白Web项目或者Web Forms项目，其中将直接提供Scaffold相关菜单。



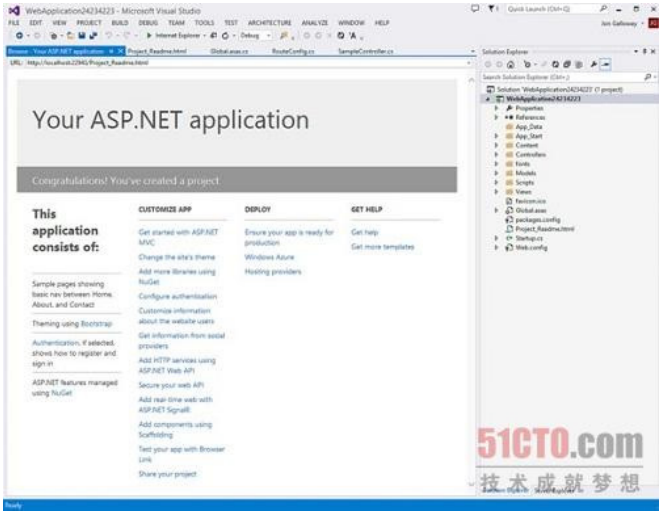
在目前的RTM版本中，我们拥有MVC 5与Web API 2两套控制器。我们曾在预览版本中对Web Forms支架进行过评测，但在RTM版本中该系统又经过了重新调整。我们期待着在即将推出的正式版本中它能迎来更多改进。

这套支架系统不仅能够作用于全局ASP.NET框架，而且还在未来可扩展性方面留下很多可能性。虽然目前的版本还不完整，但最终效果应该很快就会揭晓。

项目自述页面

自述并不是什么重要组件，但我对Visual Studio 2013的对应页面表示赞赏。在创建新项

目时，项目的根目录中会自动出现一个Project_Readme.html页面，可通过Visual Studio的内置浏览器打开。



我真心喜欢这项设定。

很久很久以前，当大家打算创建新项目时，Visual Studio只是把整套平台甩给用户、然后坐看开发者挠着脑袋不知该如何下手。这样不好。

后来我们向新项目模板中加入了大量入门指导信息，用于提示大家该接下来该做什么;不过用户必须手动将这部分引导内容从网站中删掉。这些提示信息的内容并不属于页面固有内容，所以也不太理想。

最新的自述机制借助HTML文件实现，不会对项目代码产生任何影响。如果不喜欢，大家可以随时将其删除。不过请别太过草率——它提供大量非常有用的链接，而且能够以特定格式创建自述文档、通过具体信息描述我们在项目中使用的另类设定。

备注：我很喜欢在Visual Studio内部浏览器上阅读自述信息这个主意——这比通过默认浏览器打开HTML页面酷得多也便利得多，毕竟额外弹出新窗口以及随之而来的启动时间让人烦躁。如果大家还在采用那种传统方式，最好尽快投入新方案的怀抱。理由很简单——如果某些不知名的“活雷锋”事先在浏览器里已经打开了好几十个标签页，载入过程会活活把人逼疯。等到“感谢您安装我们的Visual Studio扩展!”页面顺利打开，我一定已经在墙上撞死好几回了。总而言之，请优先利用Visual Studio中自带的浏览器打开自述页面。

ASP.NET MVC 5

ASP.NET MVC 5中的最大改动在于它不再作为独立项目类型。现在它可以与其它ASP.NET组件完美集成。

除了前面已经提到的这些常见功能(Bootstrap、模板、身份认证以及验证)，我们再聊聊ASP.NET MVC中的其它新内容。

属性路由

ASP.NET MVC现在支持属性路由功能，这要归功于Tim McCall(http://attributerouting.net网站的创始人)做出的卓越贡献。在属性路由功能的辅助下，用户可以通过为操作及控制器添加注解的方式指定路由机制。该功能支持多种复杂的定制化路由情况，而且允许大家保存自己的路由信息以及控制器操作。

控制器中包含的一项方法名为Hiding的操作，但我已经利用AttributeRouting在/spaghetti/with-nesting/where-is-wald中对其进行了配置：

```
1. ?
2. public class SampleController : Controller
3. {
4. [Route(“spaghetti/with-nesting/where-is-
   waldo”)]
5. public string Hiding()
```

```
6. {
7. return “You found me!”;
8. }
9. }
```

我在自己的RouteConfig.cs文件中添加了上述代码，并通过以下方式将其注入到其它MVC路由当中：

```
1. ?
2. public class RouteConfig
3. {
4. public static void
   RegisterRoutes(RouteCollection routes)
5. {
6. routes.IgnoreRoute(“{resource}.axd/
   {*pathInfo}”);
7. routes.MapMvcAttributeRoutes();
8. routes.MapRoute(
9. name: “Default”,
10.     url: “{controller}/{action}/{id}”,
11.     defaults: new { controller = “Home”,
   action = “Index”, id = UrlParameter.Optional }
12.     );
13.     }
14.     }
```

大家可以点击此处阅读更多关于ASP.NET MVC 5中属性路由的相关信息。

过滤器强化

过滤器新增两种类型，分别为Authentication Filters与Filter Overrides。

Authentication过滤器是ASP.NET MVC中的全新过滤器类型在ASP.NET MVC流程中优先运行，从而允许用户为每项操作、每套控制器或者全局控制器指定验证逻辑。Authentication过滤器处理请求中的证书并提供对应主体。Authentication过滤器还可以为未经授权的请求添加验证质询。

Override过滤器允许大家改变特定方法或者控制器所应用的过滤器。Override过滤器能够为特定范围(例如操作或者控制器)设定不适用的过滤器类型。这一特性允许大家在全局范围配置过滤器的同时，排除特定操作或者控制器。

ASP.NET Web API 2

ASP.NET Web API 2当中包含大量新功能。

属性路由

ASP.NET Web API支持ASP.NET MVC 5中的属性路由系统。大家可以点击此处阅读更多关于Web API中属性路由机制的介绍。

OAuth 2.0

ASP.NET Web API具备OAuth 2.0支持能力，这是由于其借助了运行在OWIN中的安全中间件(下面将进一步介绍)。这一点对于已验证单页面应用等功能来说非常重要。

OData改进

ASP.NET Web API现在全面支持OData。为了实现这一点，Web API纳入大量最强大的执行

机制：$select、$expand、$batch以及$value。大家可以点击此处阅读由Mike Wasson撰写的OData执行机制支持评论。

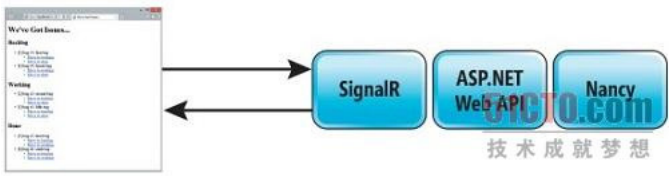
想要囊括全部功能需要一份相当庞大的列表，其中包括CORS(即跨域请求共享)、IHttpACtionResult、IHttpRequestContext等等。我想大家最好在发行说明中对它们进行进一步了解。

OWIN与Katana

我最近刚刚撰写过一篇关于OWIN与Katana的文章，我也是这两套方案的坚定拥护者。

OWIN是一款开放Web接口，专门针对.NET。我们可以将其看作类似于HTML或者HTTP的规范，因此大家无法安装OWIN。OWIN带来的好处在于，它是一套业界通行规范，因此任何采纳这套方案的组件都可以由此接入ASP.NET堆栈——无论是中间件还是主机。

Katana是微软推出的OWIN衍生方案。它利用OWIN将验证、处理、模块、IIS托管等元素串连起来，从而使ASP.NET能够托管OWIN组件与Katana组件，并使其运行在其它OWIN衍生方案当中。



Howard Dierking在MSDN杂志中撰写了一篇精彩的文章，从深层次围绕Katana展开探讨——文章题为《Katana项目入门指南》。他通过一个有趣的例子展示了以OWIN为基础且在单一堆栈中使用SignalR、ASP.NET Web API以及NancyFx组件的实施途径。

如果这样的处理方式对于大家有意义，那当然最好。如果暂时没有也别担心，但请各位务必对此保持关注。随着ASP.NET对插件容纳能力的不断提升，将有越来越酷的使用方式逐步出现。

Visual Studio Web工具

好吧，这东西相当疯狂。Visual Studio在过去几年来不断迎来多种出色的Web开发功能，但微软最终将其整理并构建起新版本仍然令人激动。

Visual Studio是我目前在Web文件领域最喜爱的代码编辑器：CSS、HTML、JavaScript以及大量流行库，一切都能打理得妥妥当当。

不要再把Visual Studio当作是一种只能用来编写后端代码的大型编辑器了。不要再利用Notepad(或者Sublime、Notepad++等)编辑HTML以及CSS。Visual Studio在配备SSD的现代计算机上只需两秒就能完成启动。在HTML属性、CSS类、jQuery或者Angular语法方面出现拼写错误实在是太愚蠢了。这不会让你成为更优秀的开发人员，只会让你变成浪费时间的傻瓜。

Browser Link

Browser Link是一种存在于Visual Studio与全部网络浏览器之间的实时、双向连接。它只在我们以本地方式进行调试时才能发挥作用，但却适用于任何一款网络浏览器，包括模拟器。

大家可能已经在一些演示中看到过如何在编辑器内部做出修改以刷新浏览器，我也承认这种方式很酷。但这真的仅仅只是开始。这是一种双向连接，而且专门为扩展性而打造。这意味着大家可以通过编写扩展将信息由正在运行的应用程序(可以在IE、Chrome甚至是移动模拟器当中)返还至Visual Studio。

Mads及其技术团队展示了他们如何在浏览器的编辑模式下将源HTML返还至浏览器。我们甚至可以在演示中看到他们如何呈现HTML执行过程、检查兼容性问题以及检查无用的CSS类等等——总之，没有做不到只有想不到。

新型HTML编辑器

早期版本中的HTML编辑器中存在大量陈旧代码，给我们的改进带来极大束缚。技术团队重新编写了HTML编辑器，从而让Visual Studio中的新型扩展功能发挥作用——这样的处理方式使他们得以将全部功能添加进来，包括CSS类与ID智能感知(只需输入style=“”即可获得一份关于当前项目的类与ID列表)、文档格式智能化排版以及自动同步JavaScript引用等等。

这是一段来自Mads Kristensen的视频教程，时长为三分钟。

集成化Windows Azure网站创建与发布

Windows Azure门户机制运作良好、与普通网站的使用方式无异，但我们必须通过额外的步骤才能通过这套门户创建新网站、而后下载并发布配置文件、最后才能将其导入至自己的站点。这就像连续进行十次点击——实在是让人感到疲劳，每次进行这项工作之后我都觉得需要睡一觉。

这一切在Visual Studio 2013的Server Explorer中都得到了更新，因此我可以直接右键点击Windows Azure节点以创建新网站。在后续的发布过程中，我可以直接将站点分布配置文件导入进来并立刻执行。

这意味着我能够利用免费的20MB SQL数据库建立起新的Windows Azure网站并将其直接在Windows Azure中发布——整个过程完全不涉及Visual Studio。这真的很棒，我很喜欢。

大家可以**点击此处**阅读《利用Visual Studio 2013 RC创建全新Windows Azure网站》一文，相信会给您带来帮助。

更多Visual Studio Web开发功能

这篇文章仅仅算是概述，没有介绍到的卓越功能还有很多，其中包括JavaScript编辑、CSS编辑、发布以及Page Inspector(用于在Visual Studio内部实时呈现您的页面)等等。点击此处可以观看更多功能演示视频。

一切只是开始，广泛的世界正等待我们探索。

好了，我们的这篇总结文章到这里就要迎来尾声了。请大家到http://asp.net/vnext中了解更多信息，并尽快下载Visual Studio 2013开始属于自己的新旅程。■

IT行业中的六个肮脏秘密

□ 核子可乐译

IT专业人士往往若无其事地传播灰色谎言、玩弄技术业务的黑色权谋

IT专业人士往往对问题背后的真正由来了如指掌——有时候，他们本人就是造成麻烦的罪魁祸首。

我们向读者朋友征集在工作中遇到过的最肮脏的IT秘闻——那些来自灰色地带的谎言以及其他人根本意识不到的技术阴暗面。整理工作结束后，我们将这些“秘密”交给相关领域的专家，让他们对其进行分析。其中一部分得到了专家的共鸣，但另一些则未受肯定。

系统管理员手中的权力足以成为CIO最恐怖的噩梦？IT员工仍然拿走企业设备？我们保存在云端的数据是否真会不翼而飞？技术支持服务开出的价码算不算高得离谱？

通过今天的文章，我们将共同了解当事者与相关专家们的观点。

IT肮脏秘密第一条：系统管理员抓着公司的小辫子

IT这只黄鼠狼居然成了数据这只肥鸡的守门者

任何一位关注过爱德华·斯诺登其人其事的朋友肯定已经了解到，一位系统管理员能够造成怎样的破坏。但即使是IT人士自己可能也无法想象，不受约束的管理员究竟拥有何等惊人的权力与危害性。

“对于IT人士而言，根本不存在秘密这回事，”安全服务托管供应商Network Box USA公司CTO Pierluigi Stella表示。“我可以在自己的防火墙上植入探测工具，从而掌握特定计算机上进进出出的任何一个数据包。我能够看到人们在消息当中写下哪些内容、他们访问了哪些互联网站、在Facebook上写了些什么。事实上，道德是惟一一种能够约束IT人士不至于误用或者滥用这种权力的因素。IT人士完全就是存在于我们身边的国安局的缩影。

这种情况相当常见，甚至大部分CIO都已经意识到了这一点，数据保护企业SafeNet公司首席战略官Tsion Gonen指出。

“我估计九成以上企业都面临这样的危机”他表示。“企业安全的可靠性与IT管理员的可信程度密切相关。我们很难准确弄清到底有多少系统管理员正在滥用自己手中的访问权限——但可以肯定的是，数量已经多到足以占据每星期的报纸头条。最可怕的是，安全风险往往来自那些负责为企业员工分配访问权限的家伙。”

数据治理方案供应商Varonis公司副总裁David Gibson也认为，管理员一般确实有能力在神不知鬼不觉的情况下进行数据访问，但他提出了相对比较具体的比例数字——50%。他补充称，这样的问题并不只发生在管理员身上——大部分用户都有能力访问超出工作范围之外的更多数据。

他表示，要想解决这一难题，方案可以归纳为两个方面：利用“最低权限”模式削弱员工的访问能力；持续对数据访问者进行监控。

“企业需要有能力查看哪些员工访问了什么样的数据，这些数据归属于谁以及谁已经访问了其中的哪些文件，”他表示。“以此为基础，IT部门将能够与数据所有者直接接触，从而在削减权限与保持可接受的使用感受之间找到平衡点。”

IT肮脏秘密第二条：员工很可能正在“以厂为家”

那些“退休”了的IT资产很可能以出人意料的方式焕发第二春

陈旧的技术设备很少真正被丢进垃圾桶，它们往往能快速找到自己的新家——有时候，“新家”与IT员工的家会产生交集。

“员工盗窃被淘汰的设备早已不是什么新鲜事了，”Retire-IT公司CEO Kyle Marks指出——这是一家专门处理与IT资产有关的欺诈与隐私合规问题的企业。“我还从来没见过任何一位从未将公物占为己有的IT从业者。对于大多数人来说，拿点业已淘汰的设备并不算什么了不起的事情。很多人也不会将此视为安全威胁——一旦设备报废，他们会将其当成可以随意处置的东西。

将设备从垃圾堆或者回收站里捡走的最大问题在于，其中很可能还包含有敏感数据。一旦这部分数据遭到曝光，很可能会给公司造成巨大损失，Marks指出。当然，即使没有什么后续麻烦，这样的行为本身也属于盗窃公司财产。

“盗窃与欺诈属于很严重的事态，极有可能引发大规模隐私事故，”他补充称。“如果任由这种态势发展下去，无法无天的IT员工很可能将整个企业推向崩溃的边缘。然而在大多数情况下，负责确保所有资产得到妥善处理（即删除所有数据）的主管人士往往是IT部门的一员。企业需要建立一套‘逆向采购’流程，以保证资产遵循正常方式退出自己的工作岗位。”

但是不是每一位IT员工都会对陈旧硬件伸出贪婪的双手？某位拒绝公布名字的资深IT资产处置从业者表示，实际情况与Marks所做出的推论相去甚远。

“我并不是说盗窃活动并不存在，”他解释道。“我只是想说明，我从未遇到任何一位对陈旧硬件抱有固定处理模式的从业者。”

他同时补充称，大部分设备之所以消失不见，主要是由于丢失或者其它一些难以说清的原因——例如被运到了错误的地点。

“这听起来有些否定一切的倾向，事实上很多企业都对自身以完全诚实可信的态度提供安全服

务、坚持处事方式的做法表示骄傲。”

IT肮脏秘密第三条：将数据保存在云中——比你想象的更危险

如果跟法律条文扯上关系，世界上任何一种安全机制都将无法保护我们

把数据存储在云环境下确实很方便，但我们也很可能需要为这样的便利付出高昂的代价：在如同一团乱麻的法律诉讼当中，我们的数据将不翼而飞。

“大多数人并没有意识到当自己的数据被保存在云环境中其他人的系统上、并与其它企业的数据比邻而居时，一旦对方遭遇法律纠纷、我们的数据也可能受到殃及而不得不被公之于众，”IT支持企业CSI集团老总Mike Balter指出。

换句话来说，大家的云数据很可能由于针对他人的调查活动而遭受意外牵连——倒霉是惟一的理由，只是因为跟嫌疑方共用了一台服务器、企业就可能遭受严重损失。

2012年1月就发生过一个典型案例，当时美国与新西兰当局关闭了Kim Dotcon公司的MegaUpload文件柜。除了确实涉嫌盗版的大量电影资料之外，当局还没收了成千上万来自守法客户的数据且拒绝归还。时至今日，这些可怜的普通用户仍然无法确定自己的数据还能不能被重新找回。

“数据遭到扣押的风险是真实存在的，”Touro法律中心商业、法律与技术研究部门主任Jonathan Ezor做出证实。“如果执法部门或者其它政府官员拥有任何法律依据，则完全可以收缴存储设备或者系统——在某些特殊事件中可能需要获得批准——而这些系统中的数据无论是否存在嫌疑，都可能遭到剥夺。总之，任何一家企业的数据在存储于控制范围之外时，都将不可避免地受到某种程度的窥探——至少他人有权访问同一套硬件设备。

要想保护自己免受这种最坏状况的影响，用户必须自己的数据到底被保存在哪里、哪些法律条文符合当前情况，云案例企业JumpCloud公司CEO David Campbell表示。

“我们的建议是寻找一家能够对服务器及数据的物理位置做同保证的云供应商，例如Amazon，这样大家才能以主动姿态控制未知风险，”他指出。

Ezor同时补充称，对数据进行加密能够有效防止获得数据的家伙成功解读其中的内容。另一个好主意是：在手头常备一份数据备份。我们真的不能确定什么时候这就成了企业的最后一根救命稻草。

IT肮脏秘密第四条：员工勒紧裤带，老板却开出空头支票

搞财务的最苦逼

对于几乎任何一家中型或者大型企业来说，采购批准流程拥有两种执行方式，Hawkthorne集团CEO Mike Meikle指出——这是一家高级管理与信息技术咨询企业。首先是官方采购流程——时耗极长、我们需要像马戏团里的小猫小狗那样钻过一个又一个审批“火圈”。除此之外，还有一条特殊的“贵宾畅行版钻石通道”，当然只供少数“特殊人物”使用。

“企业高管级别的人士都有自己的采购通

道，”他解释称。“对于那些需要花掉IT人士八个月时间的审批流程，这些高管往往在几个星期之内就能搞定——这还是非常保守的估计。我将此称为‘贵宾畅行版钻石通道’。在我所接触过的政府机关或者私营企业当中，没有一个能彻底摆脱这种只存在于背地里的神秘采购途径。”

官方流程之所以要刻意为难员工，就是不希望他们花企业的钱，Meikle指出——当然，除非他们能想办法走上这条秘密通道。他同时指出，遗憾的是CIO往往没有资格加入这个贵宾俱乐部，这意味着大型技术采购往往会在没有经过严格的成本分析或者考查IT战略方针的情况下就被敲定。

“他们会一起出去吃午饭，供应商则在他们耳边不断灌输甜言蜜语；接下来的事情大家就都知道了：几十万美元被慷慨地抛出，换来另一套移动应用管理方案——这帮家伙根本没意识到企业已经有一套这种方案了，”他愤愤不平地表示。“现在我们有两套移动应用管理方案——要这么多干啥，拿来吃吗？”

但事实并不一定如此，某位来自军方及财富百强企业的匿名人士提出反对意见。虽然很多企业确实可能回避了标准采购流程，但其中涉及的往往总是IT部门迫切需要的对象——这么做是不希望把时间浪费在繁文缛节之上，他表示。“非技术高管根本不具备制定大型采购决策所必需的IT知识，”他补充道。“如果某位高级行政人员回避了采购审核流程，执意签署采购订单并要求供应商发货，那么后续出现的一切技术失误都应该通过问责及追溯机制归结到这家伙身上。这就像氪石之于超人——是他们最大的克星。”

IT肮脏秘密第五条：在客户支持的天平中站在弱势的一端

技术人员只是在玩弄脚本而已

相信大家对这样的情景不会感到陌生：我们通过手机与相隔半个地球之外的技术支持人员进行沟通，但在寥寥数语之后我们就发现对方的技术水平根本不行、只是在对客户照本宣科。猜猜怎么着？实际情况很可能正是这样。

“IT支持是一种廉价商品，”Strive技术咨询公司总裁Tim Singleton指出——这是一家高端技术支持企业、专为中小型客户提供服务。“大部分能够帮得上忙的工具都不用花钱，计算机对使用者的知识要求也不再像过去那么严苛。我们邻居家的小女孩很可能与技术娴熟的达人一样有能力像IT企业那样解决你的计算机故障。”

但也有人认为这样的结论太过武断。虽然某些简单问题确实体现不出技术支持团队的必要性，但在复杂问题方面专业建议仍然不可或缺，企业级远程IT支持方案供应商Bomgar公司服务与支持业务高级副总裁Aramis Alvarez指出。

“将IT支持称为‘廉价商品’的不妥之处在于，我们不能对所有技术难题一概而论，”Alvarez表示。“某些基础问题确实能被熟悉技术的普通用户准确诊断，但病毒感染等更为复杂的情况则不行。邻家女孩也许真的具备不少技术知识，但她最终很可能对计算机上的数据造成严重破坏。”

最后，我们不得不为了收拾残局而付出更高代价，New York Computer Help公司CEO Joe Silverman补充道——这样的问题往往发生在企业在技术方面偷工减料或者内部IT部门负担过重的情

况下。

“在日常工作中，我们发现纽约当地有很多办公组织及职能部门在以马马虎虎的态度对待计算机维修工作或者IT岗位——从其它公司拉人、找家庭成员顶替或者招揽半吊子水平的朋友，”他指出。“有时候财务部门的员工也会跑来客串解决计算机问题，但他们往往太忙或者经验不够丰富，无法修复发生故障的硬盘、主板或者电源。如果网络或者服务器发生崩溃，大家真的打算依靠财务员工完成这项工作、还是更信赖拥有二十年从业经验的资深网络工程师？”

IT肮脏秘密第六条：我们知道的比你想象的多得多

广泛收集数据，掌握全盘信息

以为自己正处于国安局的密切监控之下？与消费者营销企业与数据代理商比起来，国安局真的只能算是小儿科。

其中最大的黑手就是赌场，前任赌场数据库经理、曾根据个人经历撰写并出版《巧取豪夺：穿越繁荣之城》一书的J. T. Mathis这样说道。“当大家走进赌场时，押下的赌注绝不只是金钱——各位其实是在用自己的个人数据碰运气。”根据Mathis的估算，他前雇主营销数据库中大约包含十万个以上活跃或者非活跃的赌徒姓名。

“从踏入赌场的那一刻起，大家的一举一动就处于追踪之下，”Mathis指出。“如果各位坐在一台老虎机前，赌场管理者会知晓你的当前位置、一共拉下过多少次手柄、投下过多少个硬币；他们知道你喜欢在4：30吃饭而且偏爱龙虾拼盘。他们清楚你最喜欢的香烟与酒水品牌、知道你是否在房间内观看色情节目。

而在你暑期再度光临时，他们能够一眼洞悉随你一同前来的并非原配妻子，这样员工才能叫对她的名字‘Cindy’而非你太太的名字‘Barbara’。”

前任赌场高管、现就职于路易斯安那州立大学的Michael Simon教授证实了Mathis的说法。但他补充称，赌场收集数据的方式与CVS、PetSmart或者Amazon相比并无多大区别。

“我现在带的MBA班主攻数据库分析与发掘方向，我们调查过的所有企业都有收集客户信息并根据客户个人习惯提供服务的行为，”他表示。作为《我的游戏人生：前任赌场高管的个人视角》一书的作者，Simon补充称“这已经成为当下的常规商业惯例，而绝非什么不可告人的秘密。举例来说，我带着自己的狗到PetSmart享受特殊产品与服务，他们提供的项目一定符合我的个人消费习惯——我对此感到非常满意。

从另一个角度看，PetSmart其实在以非常高效的方式提供我想要的东西、而不是浪费时间为我准备根本不可能接受的东西——例如折扣猫粮或者热带鱼。”

但只有一件事有所不同：Mathis于2012年5月被赌场裁掉，但他当时手头仍持有数据库副本。而在尝试将副本还给赌场方面时，他的运气实在有些糟糕——对方拒绝接听他的电话——既然这样，他只好用手头的数据谈谈关于赌博的那些事儿了。■

对于云计算及移动这两大新兴领域，企业IT部门必须找到新的处理方式——因为在这里他们不再拥有至高无上的掌控权。在多数企业中，管理者们开始意识到为了保持自身在市场中的竞争力，他们需要为客户部署各类Web及移动应用并观察其实际效果。

IT仍在迅猛行： 2014年之后的九大发展趋势

谁说企业IT枯燥无味？卓越的思维与大胆的技术新风向正以前所未有的主动性改变一切，现在是时候考虑由此引发的长远影响了。

精彩的2013年尚未结束，但我已经迫不及待要把自己对于这么多新兴发展趋势的感受与大家分享。为此，我准备了这样一份对未来一年的前进展望。我原本以为自己会把滔滔不绝的预测结论作为主要内容，但在今年这样一个历史性时刻，众多趋势已经开始酝酿并初步结出果实——我敢肯定，其中大部分都将在未来的十二个月中继续保持这一良好势头。

下面就是我总结出和九大技术发展趋势，如果各位朋友拥有自己的主张、不妨在评论栏中与大家分享。

1. 云计算成为新的硬件形式。Pivotal公司CEO Paul Maritz显然对这一结论深表赞同。他的论点是：各大行业都已经开始在新型计算平台的推动向走向变革，从PC到客户-服务器再到互联网皆是如此。

在服务器、存储以及网络领域，设备的运作方式已经经过融合而形成类似于一整套大型“机器”的体系，在这里应用程序能够拥有极致可扩展性、基础设施必须与虚拟化相结合并拥有中央控制特性——而这，就是我们所说的“软件定义”概念。最终，这一趋势将超越SDN（即软件定义网络）的单方面诉求，开始将数据中心内的每一套系统涵盖进来——就连空调系统也不例外。由公共云供应商提出的高级软件控制规划将继续逐步进入企业环境。

2. 互动系统正当其时。我们要如此夸张的规模化云扩展能力干嘛？不是为了ERP这类老式企业“记录系统”，因为在这类系统中数据模型很少改变，我们也大概了解会有多少客户对其加以运用。云计算巨大力量的主要服务对象在于“互动系统”：面向客户的Web及移动应用，它们的普及将给整个市场带来不容忽视的震动效果。

对客户互动效果进行优化已经成为目前最热门的技术领域，由此产生的推动力也成为弹性基础设施、新型数据库技术以及收集并分析大数据（主要是Web点击流及其它用户数据）的发展源头。利用Hadoop类应用实现的大数据分析机制很可能成为过去十年以来最具代表性及划时代意义的单项企业技术进步。

紧随其后的则是NoSQL数据库（例如

MongoDB、Cassandra以及Couchbase），其向外扩展能力几近疯狂、采用的数据模型也在不断增加。

3. 大数据一马当先。大数据分析为我们带来诸多极具吸引力的承诺，但在短期之内具体大数据解决方案数量太多、需要解决的问题则相对较少。从长远角度看，大数据的潜力绝不仅限于对电子商务体系进行优化，而必定会融入各类垂直行业——从制造业到运输业再到电网体系，可谓无所不包。

但要让垂直领域享受大数据带来的收益，首先需要实现工业互联网普及（也就是我们常说的‘物联网’）。我们不妨想象这样的情景：无数联网传感器提供大量远程测量数据，从而改进产品设计、准确预测故障。通用电气与IBM在这一领域取得了早期领导地位，但我们目前尚处于起步阶段。

从现在开始，未来多年内工业互联网将以燎原之势迅速铺开，到那时大数据将真正变得非常非常庞大，而对大数据分析方案的渴求也将前所未有地强烈。与此同时，如果在2014年中某些技术泡沫最终破灭，那么大数据也将首当其冲。

4. 云集成脱颖而出。大数据由于自身规模所限而缺乏迁移便利性，因此现在越来越多的数据被直接保存在云端以提供给云分析机制进行处理。在一般情况下，云方案，尤其是亲自打理数据存储工作的SaaS应用，有可能让企业用户重蹈旧日那种孤立隔离的覆辙。

换句话说，各产品之间的实际差异非常小，但客户记录却散落在彼此隔离的不同数据存储体系当中（连同那些本应加以共享的宝贵处理流程）。

要解决这一难题，摆在面前的有两个答案：云集成以及更多、更好的API。云集成方案比比皆是，其中较为知名的有Cordys、戴尔Boomi、IBM Cast Iron、Informatica、Layer 7、MuleSoft、SnapLogic以及WSO2等。另外，API显然也拥有自己的联盟——例如由Apigee提供的新型API解决方案，这套方案允许企业公布并维护自己的公共API。

5. 身份验证代表着新的安全机制。尽管这种说法有些夸张，但事实上身份验证如今已经成为内部及SaaS应用当中不可或缺的重要组成部分。管理谁曾经访问过什么内容——并在员工离开公司之前取回这些内容——正变得越来越重要、同时也越来越复杂。

微软、Okta、Salesforce以及其它多家厂商已经开始推出解决方案。没有云验证管理的辅助，企业根本无法安全高效地实施公共云方案。

6. 内存成为新的存储机制。内存容量的扩大体现在两个方面。在软件方面，每一家关系类数据库供应商都在为自己的产品添加内存内功能，这主要是为了满足分析任务的需要并显著降低大型处理工作所消耗的时间。

而在硬件方面，以PernixData为代表的各类方案开始在服务器当中利用闪存打造容量更大的分布式缓存，从而大大降低必须指向SAN的读取与写入操作的比例。

7. 通向未来的道路由JavaScript铺就。由于各类移动设备纷至沓来（包括新型电视及汽车等），我们正面临着一个全新时代——在这里客户端硬件正呈现出前所未有的多样性。没人愿意为每一种平台维护一套独立的原生客户端应用。

如果大家希望能让自己的维护清单上只有一套代码库，那么应用就必须运行在浏览器之上——换言之，我们的产品必须属于JavaScript/HTML5应用。

这就难怪几乎每周都会有新的JavaScript框架出现，Famo.us等网站也会不断推送消息、向我们传达JavaScript的最新创造性使用方式。另外，跨平台移动开发环境（例如PhoneGap）也允许大家轻松将JavaScript应用转化为原生应用。

8．企业开发人员转向PaaS。到目前为止，PaaS的主要客户群体为商业软件开发商以及专业服务厂商。不过随着越来越多的企业推出自家Web及移动应用程序，内部开发人员也将逐步发现PaaS方案所带来的便利——目前市场上的主要选项包括微软Azure、Pivotal Cloud Foundry、红帽OpenShift以及SalesForce Heroku。它们全都提供灵活的编码机制、测试环境以及在云中部署应用程序的能力。

IBM今年对Cloud Foundry的全力推动可算一大重要里程碑——此举可能让不少企业用户打消疑虑、放心将自己的代码安置在其它厂商的平台之上。此外，IDC公司也做出预测，称针对特定行业的PaaS产品（包括针对特定垂直行业的预置服务）将迎来崛起。

9. 开发人员继续主宰一切。如果要在以上各项预测当中找到一条共同线索，那就是Marc Andreessen在两年前提出的论断再次得到验证——软件正吞噬世界。由于存在这么多需要为之编写代码的平台——如今甚至数据中心基础设施本身也开始具有可编程化特性——现有开发人员数量肯定无法满足如此庞大的需求。招聘市场上给出的薪酬与职位越来越高——至少对那些具备理想技能组合的求职者是如此。我们不禁要问：如何才能更好、更高效地培养出拥有这些技能的专业人才？

这九大趋势代表着短时间内即将出现的巨大变革。至少我们可以说这些趋势将给IT行业带来有趣的影响。

总结起来，如果软件开始定义基础设施，那么硬件就将变得更加商品化——其中包括网络设备。如果应用程序的发展方向日趋明朗，即我们终将编写出能够运行在任何客户端设备上的应用，那么针对运行平台的选择将不再重要。

与此同时，软件的交付方式也发生了永久性改变。IBM、甲骨文、SAP以及其它常常植根于企业用户环境内的传统供应商可以继续保持自己的竞争优势，但除此之外还有什么令人兴奋的新内容？主要备选答案可能在于开源、SaaS或者成本低廉的移动应用。在我看来，虽然很多新兴企业被收购，但大部分从业厂商可能需要重新调整其收益预期。

对于云计算及移动这两大新兴领域，企业IT部门必须找到新的处理方式——因为在这里他们不再拥有至高无上的掌控权。

在多数企业中，管理者们开始意识到为了保持自身在市场中的竞争力，他们需要为客户部署各类Web及移动应用并观察其实际效果。IT部门能否及时获得相关技能以满足由上述趋势带来的需求？或者说企业管理层是否有可能转而向SaaS方案、敏捷开发厂商或者其它外部供应商寻求帮助，而将IT部门打入冷宫？答案仍未揭晓，我将拭目以待。■

Spring MVC+JQuery+Google Map打造IP位置查找应用

□ 廖煜嵘译

在本文中，读者将学习到如何使用Spring MVC框架和jQuery及Google Map,制作一个简单的根据IP位置查找应用。用户可以在页面中输入一个IP地址，然后通过Google Map显示该IP所在的大概地理位置（注：本文使用的数据库是GeoLite）。本教程完成后，其效果图如下：

在本教程中，用户将用到如下技术：

- Spring MVC frameworks
- jQuery(Ajax Request)
- GeoLite 数据库
- Google Map

其中用户的操作步骤如下：

- 用户输入IP地址,然后点提交按钮
- jQuery发出Ajax请求到Spring MVC中的控制器
- 在Spring MVC控制器中处理并返回JSON格式数据
- jQuery处理返回的json数据并通过Google Map展示给用户

1 项目结构

项目结构如下图,使用的是MAVEN工程

这里,我们要下载专门的地理数据库GeoLite,其中我们下载的是GeoLite格式的IP数据库,地址如下:http://dev.maxmind.com/geoip/legacy/geolite/,并放置在resource文件夹下。

2 MAVEN包依赖

在本项目的pom.xml中，加入如下的包依赖，如下：

```
1. //...
2. <properties>
3.     <spring.version>3.2.2.RELEASE</spring.version>
4.     <maxmind.geoip.version>1.2.10</maxmind.geoip.version>
5.     <jackson.version>1.9.10</jackson.version>
6. </properties>
7.
8.  <dependencies>
9.
10.     <!-- Spring Core -->
11.     <dependency>
12.         <groupId>org.springframework</groupId>
13.         <artifactId>spring-core</artifactId>
14.         <version>${spring.version}</version>
15.     </dependency>
16.
17.     <!-- need this for @Configuration -->
18.     <dependency>
19.         <groupId>cglib</groupId>
20.         <artifactId>cglib</artifactId>
21.         <version>2.2.2</version>
22.         </dependency>
23.
24.         <dependency>
25.         <groupId>org.springframework</groupId>
26.         <artifactId>spring-web</artifactId>
27.         <version>${spring.version}</version>
28.         </dependency>
29.
30.         <dependency>
31.         <groupId>org.springframework</groupId>
32.         <artifactId>spring-webmvc</artifactId>
33.         <version>${spring.version}</version>
34.         </dependency>
35.
36.         <!-- ip to server location -->
37.         <dependency>
38.         <groupId>com.maxmind.geoip</groupId>
39.         <artifactId>geoip-api</artifactId>
40.         <version>${maxmind.geoip.version}</version>
41.     </dependency>
42.     <!-- Jackson -->
43.     <dependency>
44.         <groupId>org.codehaus.jackson</groupId>
45.         <artifactId>jackson-mapper-asl</artifactId>
46.         <version>${jackson.version}</version>
```

```
47.     </dependency>
48.     </dependencies>
49.     //...
```

3 Spring MVC+Geo Lite

下面首先编写根据IP查找地理位置的接口,命名为ServerLocationBo.java,代码为:

```
1. package com.mkyong.web.location;
2. public interface ServerLocationBo {
3.     ServerLocation getLocation(String
   ipAddress);
4. }
```

其实现类的代码为:

```
1. package com.mkyong.web.location;
2.
3. import java.io.IOException;
4. import java.net.URL;
5. import org.springframework.stereotype.
   Component;
6. import com.maxmind.geoip.Location;
7. import com.maxmind.geoip.
   LookupService;
8. import com.maxmind.geoip.regionName;
9.
10.     @Component
11.     public class ServerLocationBoImpl
   implements ServerLocationBo {
12.
13.         @Override
14.         public ServerLocation
   getLocation(String ipAddress) {
15.     String dataFile = “location/GeoLiteCity.
   dat”;
16.     return getLocation(ipAddress, dataFile);
17.     }
18.     private ServerLocation getLocation(String
   ipAddress, String locationDataFile) {
19.
20.      ServerLocation serverLocation = null;
21.      URL url = getClass().getClassLoader().
   getResource(locationDataFile);
22.       if (url == null) {
23.     System.err.println( “location database is
   not found - “
24.     + locationDataFile);
25.       } else {
26.      try {
27.     serverLocation = new ServerLocation();
28.     LookupService lookup = new
   LookupService(url.getPath(),
29.     LookupService.GEOIP_MEMORY_CACHE);
30.     Location locationServices = lookup.
   getLocation(ipAddress);
31.
32.     serverLocation.
   setCountryCode(locationServices.
   countryCode);
33.     serverLocation.
   setCountryName(locationServices.
   countryName);
34.     serverLocation.setRegion(locationServices.
```

```
region);
35.     serverLocation.
  setRegionName(regionName.
  regionNameByCode(
36.     locationServices.countryCode,
  locationServices.region));
37.
38.     serverLocation.setCity(locationServices.
  city);
39.     serverLocation.
  setPostalCode(locationServices.postalCode);
40.     serverLocation.setLatitude(
41.     String.valueOf(locationServices.latitude));
42.     serverLocation.setLongitude(
43.     String.valueOf(locationServices.
  longitude));
44.      } catch (IOException e) {
45.     System.err.println(e.getMessage());
46.      }
47.              }
48.              return serverLocation;
49.             }
50.     }
```

在上面的这个方法中,在getLocations方法中,加载了GeoLite格式的IP地址库文件GeoLiteCity.dat,并且调用getLocation方法进行IP的查找,.在getLocation方法中,通过GeoLite创建一个LookupService类的实例，其中传入要查询的IP地址库文件，然后再调用其getLocation方法进行查询，这里查询出来的结果构造成serverLocation对象，下面来看下其serverlocation对象的代码:

```
1. package com.mkyong.web.location;
2.  public class ServerLocation {
3.
4.      private String countryCode;
5.      private String countryName;
6.      private String region;
7.      private String regionName;
8.      private String city;
9.      private String postalCode;
10.             private String latitude;
11.             private String longitude;
12.
13.             //getter and setter methods
14.     }
```

然后我们使用Spring MVC框架中的Jackson对结果进行转换，转换为json，代码如下:

```
1. package com.mkyong.web.controller;
2.
3. import org.codehaus.jackson.map.
   ObjectMapper;
4. import org.springframework.beans.
   factory.annotation.Autowired;
5. import org.springframework.stereotype.
   Controller;
6. import org.springframework.web.bind.
   annotation.RequestMapping;
7. import org.springframework.web.bind.
```

```
annotation.RequestMethod;
8. import org.springframework.web.bind.
annotation.RequestParam;
9. import org.springframework.web.bind.
annotation.ResponseBody;
10.  import org.springframework.web.servlet.
ModelAndView;
11.  import com.mkyong.web.location.
ServerLocation;
12.  import com.mkyong.web.location.
ServerLocationBo;
13.
14.  @Controller
15.  public class MapController {
16.
17.  @Autowired
18.  ServerLocationBo
serverLocationBo;
19.  @RequestMapping(value = "/",
method = RequestMethod.GET)
20.
21.  public ModelAndView getPages() {
22.  ModelAndView model = new
ModelAndView("map");
23.  return model;
24.  }
25.
26.  //return back json string
27.  @RequestMapping(value = "/
getLocationByIpAddress", method =
RequestMethod.GET)
28.
29.  public @ResponseBody
30.  String getDomainInJsonFormat(@
RequestParam String ipAddress) {
31.  ObjectMapper mapper = new
ObjectMapper();
32.  ServerLocation location =
serverLocationBo.getLocation(ipAddress);
33.  String result = "";
34.  try {
35.  result = mapper.
writeValueAsString(location);
36.  } catch (Exception e) {
37.
38.  e.printStackTrace();
39.  }
40.  return result;
41.  }
42.  }
```

这对于熟悉Spring MVC的用户应该并不陌生。经过转换后的结果转换为字符串。

4 JSP+jQuery+Google Map展示最后结果

最后我们在页面中，通过jQuery发送ajax请求调用Spring MVC控制层，然后将返回的结果展示在页面中，代码（略，请进入文章页阅读）■

本文的代码可以通过如下地址下载：
http://www.mkyong.com/wp-content/uploads/2013/10/SpringMvc-jQuery-GoogleMap.zip

惊喜！Java为服务器端Web应用带来最高运行速度

在基准测试中，Java基础框架以出色的性能表现傲视群雄——但真正吸引开发人员的因素却并非性能。

骨文公司似乎正持续将Java推向各个领域，从“物联网”等新兴领域到传统范畴，而且这一推动并非依靠宣传与炒作。在最近的一项基准测试当中，Java力克众多服务器端Web框架竞争对手、一举拿下最佳性能表现桂冠。不过单靠出色的性能是否足以拉拢非Java技术人员的拥护?

自2013年3月开始，软件开发企业TechEmpower公司已经对当下流行的十多种Web应用程序服务器框架进行过性能基准测试，其中包括Ruby on Rails以及Django。每一轮成功测试在参考发布在GitHub上的开源基准测试之外，还会征求来自技术社区的反馈意见。有兴趣对自己使用的框架进行基准测试的朋友可以利用同样的开源代码亲手进行检验，并将结果提交给该公司。

TechEmpower公司的第七轮基准测试结束于今年十月末——其中包括84套框架以及约200种不同的测试排列——最终结果表明，在各类领域中表现最出色的框架普遍以Java为基。其中最突出的框架有Gemini、Grizzly（旨在使Java的New I/O API更加易用）、Undertow以及Vertx。

更令人惊讶的是，不少知名度极高的框架——例如针对Ruby的Sinatra、各类ASP .Net框架以及针对Python的Django等——在性能表现方面与优胜者相比简直差了一个量级。作为块领域的新贵，Node.js确实带来令人印象深刻的性能表现，但其最终也只达到第一名的四分之一到三分之一。

甲骨文一直在为Java摇旗呐喊，并将其描述为一套可应对各类需求的多合一解决方案（把“各类”换成“全部”可能更为确切）——而其最理想的发挥舞台就是Web服务。甲骨文为Java 8制定的计划包括统一Java各个版本，从而使开发人员能够更轻松地针对不同嵌入式设备及服务器编写代码——如果ARM真的能够入主服务器平台，那么嵌入式设备与服务器之间的界线将更加模糊。另外，Java还得到Avatar项目的有力支持，这是一套JavaScript与HTML 5服务层，旨在让Java能够与Grizzly顺利协作。

如果Grizzly以及其它Java框架同胞的表现确实有这么出色，这是否意味着开发人员都会为之所吸引并尝试将一切利用Java进行重新编写？答案恐怕是否定的。尽管甲骨文很喜欢这种思维方式，但开发人员在选择框架时往往还需要在性能之外考虑多种其它因素——例如易于开发、靠谱的发布时间、开发人员亲和力以及第三分技术社区的参与程度等，这一切都会起到重要的参考作用。随便举个例子：CakePHP在TechEmpower的性能竞赛中表现糟糕，基本上与CodeIgniter和Symfony处于同一水平，但PHP及其框架把持着Web领域35%的份额，因此其人气依旧高涨。

Java框架也许确实拥有甲骨文一直在鼓吹的出色性能，但其它框架则拥有另一种形式的竞争优势，例如开发人员基础、渗透能力以及用户忠诚度等——甚至有些用户会单纯因为讨厌被甲骨文窥视的感觉而拒绝Java。■

逐利无罪：利用开源赚钱的九项秘诀

■ 在今天的文章中，我将向大家推荐九种企业利用开源赚取利润的途径。尽管其中一些方法可能会对心怀天下、希望普渡众生的热血青年造成心理打击，但我建议各位能以“在做好事之前、先把自己的事情做好”为出发点进行理解。繁荣的商业体系对项目的支持效果远远超过一堆代码，毕竟面对前途未卜的未来，单靠慈善之心还不够。总而言之，缺乏稳定性的自由可以说一文不值。

低成本营销、讨价还价、对竞争对手展开阻击——开源行业不再只与“免费”挂钩，暴利已经随处可见。

在刚刚诞生的初级阶段，开源软件就像是公益之心带给这个世界的一份圣洁礼物。程序员们努力工作，然后将自己的劳动成果交给任何一位能够从中获得帮助的用户手中。这是一种纯粹的慈善行为，每个人都将从中受益。

然而随着时间的推移，不少企业开始意识到自己完全可以从开源身上掘取利润，并像以往那样把软件当作产品进行出售。如此一来，他们既做了好事、也得得到应有的回报。对于某些早期开源倡导者来说，这样的思维方式并不会造成冲击——相反，这是非常正常的发展 方向。开源行业的领军人物之一Richard Stallman就认为“言论自由”要比“免费啤酒”重要得多。他一直秉持着这样的思路：企业可以对开源成果进行任何调整——只要他们不剥夺普通用户修改代码并发布成果的权利就行。

很多企业把Stallman的言论视为凭借开源赚取利润并改变自身命运的神圣祝福。少数最聪明的家伙想到可以利用开源项目增强自身业务、推广自家品牌并扩大企业在行业中的影响力。从这时开始，开源已经不再像过去那样属于纯粹的慈善行为——它开始变成另一种营销手段，并通过异于以往的方式挤进了商业市场。

最精明的开源拥护者们支持这种对自我利益的肯定。他们认为遵循这样的思路，则每一位参与者都有理由为开源项目做出自己的最大贡献。他们解释称，开源领域的技术天才将帮助我们控制自私与贪欲、进而将其转变为可让每个人受益的成果。项目贡献者彼此平等，而且也很少对于代码块的所有权展开争论。分享的精神使每个人都专注于软件的质量，而不再纠结于授权问题。

在今天的文章中，我将向大家推荐九种企业利用开源赚取利润的途径。尽管其中一些方法可能会对心怀天下、希望普渡众生的热血青年造成心理打击，但我建议各位能以“在做好事之前、先把自己的事情做好”为出发点进行理解。繁荣的商业体系对项目的支持效果远远超过一堆代码，毕竟面对前途未卜的未来，单靠慈善之心还不够。总而言之，缺乏稳定性的自由可以说一文不值。

开源盈利战略第一条：利用开源进行低成本营销

打广告要价不菲、办展览烧钱神速，对于企业来说，营销预算永远极度吃紧。这时，很多企业开

始将目光投向开源代码，这绝对是一套理想而廉价的宣传方案。以开源形式发布全部或者部分产品不仅能够成功吸引到用户的注意，同时还可以帮助他们了解产品的实际用途。既宣传了产品、又吸引了客户，有了这样良好的宣传基础，营销团队能够把更多精力放在与销售活动相关的后续工作身上。

某些开源企业，例如MySQL，曾经明确表示，把关注重点放在有多少用户能够免费享受自己的产品上是个严重的错误。一般来说，企业不会到处宣扬自己的用户中有九成以上根本没付过钱——理由很简单，由于开源软件包的发布成本极低、向更多用户提供产品根本不会产生多少额外支出。

从开源产品中赚取利润的诀窍在于，确保自己打算用于收费的功能具备足够的用户吸引力，并借此为产品的其它组件提供经济支持。收费的部分在整款产品中所占比例不能太大，但一定要拥有充足的理由让目标客户掏出钱来。有时候这可能是一项额外功能，例如保证那些关注稳定性的企业客户能够获得全天候的软件流畅运行效果。还有一些厂商要求一部分用户以匿名方式帮助其推广工作成果，作为回报，用户能够获得产品的开源版本使用权。这些小技巧如今已经被全世界成千上万企业所采用。

开源盈利战略第二条：利用开源代码降低支持成本

遇上难题？这有一大堆代码，自己找答案吧。

虽然这听起来有些不尽人情，但很多开源企业确实会在技术支持工作中直接向用户提供与问题对应的源代码。那些专有型企业需要编写复杂的描述文档来解释API的实际作用，但开源企业只需要将经过解码的API扔到网上就行了。任何人都有权阅读这些源代码——而且大部分人也确实会读读看。时至今日，很多企业都会为自己的软件产品设立一套技术支持论坛，难题在这里能够更快得到解决。

很好，拥有良好说明文档的开源软件将成为每一位参与者的宝贵财富。有了这笔财富，客户将有能力自己动手解决问题，而不必再坐等技术支持人员一点点发掘代码内容。支持团队能够免去亲手将代码翻译成英文的麻烦，因为这些工作完全可以交给充满活力的技术社区来完成——每个人都能从开源中受益，标准的大团圆结局。

开源盈利战略第三条：利用开源降低开发成本

您的企业需要一款工具、资源库或者组件，但组织内部开发实在成本高昂、难以承受。现在，只要大家脑子稍微一动、往开源上想想，项目就相当于已经完成一半了。花钱雇人添加必要功能也许太过愚蠢、太过慷慨，因为开源许可规定所有相关成果都必须拿出来同大家分享。但换个角度看，开源开发也能帮大家节约一半支出。如果这款软件并不是业务流程中的关键性组成部分，那么利用开源很可能是一种聪明的省钱方式，同时也能彰显我们宽宏大量、热心公益、悲天悯人的济世情怀——何乐而不为呢？

某些公司会通过向自己了解并信任的开发人员支付报酬来解决上述难题。另一些则利用自身的宣传影响力替参与项目开发的人员造势。目前，以BountrySource为代表的一些众包网站允许用户自行筹集资金雇佣程序员开发代码。这些程序员与项目本身并无关系，搞定自己的工作后带着钱离开，但他们的成果却将长久留存。

在某些情况下，多家企业可以团结起来共同开

发同一套开源代码库，这样每家公司在做出贡献时需要支付的成本就低得多。他们在节省资金的同时也打造出一套关键性工具，且每位参与者都有权加以使用——这实在是种经济实惠的妙招。即使只找到一家合作伙伴，开发成本也能立刻被削减一半。如果是十家企业组成联盟，那么成本就只相当于整个项目支出的十分之一。

开源盈利战略第四条：利用开源代码打击竞争对手

当谷歌刚刚推出其Android操作系统时，苹果的iPhone在智能手机市场上正占据着压倒性的数量优势。然而Android作为一款开源平台，能够保证谷歌与其它手机制造厂商更轻松地携手合作、从而创建起蓬勃发展的健康环境。每个人都可以使用来自这个开源联盟的应用产品。开源许可使每家公司都以平等的参与者身份访问项目、获取源代码并加以控制。他们选择Android的同时也相当于选择了安全的未来，因为他们知道谷歌不会放弃这个生机勃勃的项目。

这种共享式流程正变得越来越普遍。作为由Rackspace提供赞助的项目，OpenStack允许小型云厂商汇聚起来使用一套通用型平台，其吸引力远远超过目前市场上占统治地位的Amazon云。客户不仅能够从多家厂商当中随意做出选择，同时也可以在自己的数据中心内部安装云工具。所有围绕OpenStack建立起来的云体系都采用同样的基础结构，而且同一套脚本也保证可以在任何环境下正常运行。

开源盈利战略第五条：利用开源资源创造竞争者

开源许可让一件事情变得更加简单：创造一位竞争者。从零开始建立一家新企业的过程中，我们只需要访问源代码库并从中寻找可资利用的数据即可。在下载完成之后，大家可以直接将其整理为方案并马上以竞争者的姿态向行业前辈叫板——没错，这一切都可以分分钟搞定。

不过创造一位竞争者与为之持续提供技术支持有着本质不同。下载代码毫无难度，但让自身拥有基础技能则需要耗时数月。要想成为真正的专家，整个过程甚至会持续数年。真正的竞争者意味着建立起一个技术团队，且有能力为用户提供真正的专业知识。

正因为如此，此类根基不牢的竞争者才仅仅出现在那些供给远低于需求量的领域当中。几年之前，当Hadoop引发技术行业的广泛关注时，新兴企业开始如雨后春笋般大量涌现。每家公司在寝阶段都采用同样的Hadoop核心，但随着时间的推移，他们很快开始提供独此一家、别无分号的特殊附加方案。

开源盈利战略第六条：利用开源在市场上保持竞争力

开源世界的竞争是一条双行道。尽管任何人都可以在几秒钟内参与进来并获得源代码，但他们通常需要在许可的约束下将自己的全部创新成果作为贡献回馈给开源项目。如果新兴竞争者水平高超，那么所有老牌强队也将能够获取前者带来的研发杰作。以GPL为代表的不少主流许可要求每位参与者都必须彼此分享技术果实。

这种除了共享还是共享的规则使新兴参与者很难真正与现有领导者相抗衡。这些领导者能够轻松获得全部由后起之秀带来的创新方案，而创新者则享受不到太多成果本身带来的收益。这样的规则在

简化了竞争者出现的机制之余，也使他们几乎无法在竞争当中蓬勃发展。

作为早期开源推动者Cygnusr创始人，Michael Tiemann曾经颇有先见之明地表示：“非常幸运，开源模式再次派上了用场。除非新兴竞争者能够与我们旗下由一百多位工程师——其中大部分是受支持软件的主要开发者或者维护者——所组成的技术团队相对抗，否则他们无法取代我们‘GNU真正根源’这一稳固地位。即使从最乐观的角度讲，他们也只能通过添加增量功能让客户掏钱。不过由于软件本身的开源属性，他们所创造的所有价值都将反映在Cygnus这里。”

虽然这样的言论听起来像是来自邪恶的垄断者，但其中也并非毫无破绽。如果当前开源领导者的工作做得不好，把资金投入到毫无意义的功能强化或者挥霍到毫无附加价值的方面，新兴竞争者完全可以找机会取而代之——这并不是不可能的。

另外需要强调的是，如果有正当理由支持同一套代码基础分别存在于两种方案当中，那么开源许可将无法制约后起之秀的发展。例如，同一款软件具备两种完全不同的使用途径，那么两个团队能够轻松将彼此的业务重点区分开来。总而言之，只要方案能够指向另一个截然不同的竞争市场，那么新秀与老将之间就不再是势不两立的对手关系。

开源盈利战略第七条：利用开源进行讨价还价

虽然不少开源许可都相当灵活，但其中一些正变得愈发严厉。作为其中最新一项许可，Affero GPL坚持认为只要代码被运行在公共服务器上、那么这些代码就必须实现共享。这项严厉许可的出台是由于在过去一段时期，开源行业发现某些开发商尝试从开源软件中获益、但却逃避为其做出贡献。他们拒不向软件提供“贡献”，却恬不知耻地加以运行——有鉴于此，GPL要求参与者只能在做出“贡献”之后才有资格获得共享权利。

某些开发商认为这样的要求并不难做到。他们可能只是尝试或者提供一些免费服务，分享自己对软件的改进但前提是这些改进并没有关键到会让自己推动竞争优势。但已经有越来越多的企业发现跟规则绕弯子比直接购买商业许可更麻烦。开源许可正以强大的力量推动参与企业走向产品支持道路。

Affero GPL已经成为众多新兴项目的理想选择，其中包括NoSQL数据存储方案。以MongoDB为例，它就为自己的核心工具——数据库搭配了该许可。不过该公司选择了条件更为宽松的Apache许可对驱动程序加以保护，旨在鼓励人们更积极地与其核心产品进行对接。

开源盈利战略第八条：利用开源开发共享式标准

每一种业务、每一类市场都需要一套执行标准，从而帮助客户建立正确的期望、帮助企业了解该交付什么样的产品。开源代码往往能在建立这些标准的互操作性方面帮上大忙。

以HTML为例，我们利用这种语言来标记网络上的文档——但这项至关重要的标准也是Web浏览器行业竞争的根本性基石之一。一旦整个行业承认HTML标准的核心地位，那么浏览器厂商就能够在功能而非内容方面做出创新并进行竞争。另一方面，内容提供方则需要保证自己生成的Web页面能够在所有可用浏览器上正常显示。

开源工具往往与处于不断发展态势下的标准密不可分。以移动浏览器市场为例，苹果公司率先创

建出WebKit渲染引擎，并使其在很大程度成为移动浏览器的定义与标杆。然而最终使其发扬光大的却是谷歌及其它Android厂商。苹果可以继续掌握这项技术的专利，但这意味着iPhone与其它智能手机之间的互操作性将受到严重制约，或者在每一次互操作时都需要将网页（而且只是少数内容不多的网页，大部分正常页面无法实现翻译）通过翻译转化成经过渲染、可为其它智能手机所读取的内容。这很可能对移动市场的拓展造成严重的负面影响。因此，苹果选择将其作为开源工具包推向公众，从而借助各市场竞争者之力将其打造成一套通用型标准。

开源盈利战略第九条：利用开源掌控未来

已经有众多企业，有大有小，开始鼓励自己的全职员工从事开源项目。有些公司甚至把花费大量资金创建的代码直接贡献给开源项目。为什么要这么做？这是为了确保自己在开源项目的代码基础中拥有足够的影响力，而实现这一目标最简单的办法就是贡献代码行。

这种影响力永远不会消退。所有重大项目，例如Linux，的大多数重要贡献者如今已经被证实都受到了所在公司的授意。当然，最终目标在于确保开源代码仍然与企业自身的发展方向相吻合。如果资源库或者工具不断增长，新功能有可能同企业的专有工具之间出现兼容性问题。但如果该公司在新功能当中贡献了大量代码，则有能力确保最终成果适应自己的业务需求。正如Alto发明者Alan Kay所说，“掌控未来的最佳方式就是亲手把它发明出来。”■

链 接

编程的六月定律

上周，我被迫对一个很老的项目做一些修改。麻烦是，当开始着手时，我真的记不清这个项目究竟有多老了。

这实际上是我使用Codeigniter实现的第一个MVC项目。打开项目文件后，很多东西都让我头晕。首先，没有版本控制，第二，没有注释。

读起代码，我的“F*CK/分钟”的值一直冲破屋顶。

项目里面的Model很少。Controller层有大量重复的代码，View层肥大的令人毛骨悚然。我相信View层里的逻辑实际上比Model层和Controller层的加起来都要多。

我该为此感到羞耻吗？

答案是NO。(如果是的话我也不会写这篇博客里。)

为什么不？

因为有个六月定律。六月定律说的是，每个程序员都应该回头看看自己6个月前写的代码，并且应该会唾弃当时写的那些代码。

这就引出了本文的重点：如果你是个程序员，当你看6个月前写的代码时，如果发现跟现在写代码的水平一样，请别写了，你应该学习一些新东西了。

这就是为什么当我看到以前的代码写的奇丑无比时反而很高兴的原因。非常高兴。这说明我进步了。所以，与其为那些丑陋的代码感到羞耻，不如高兴的接受它们，这意味着你在成长。■

十二项炒作缠身却尚未证明实力的新兴技术

□ 核子可乐/译

在技术领域，已经有无数次“下一波浪潮”不断涌现、但却最终只沦为“下一波退潮”。这往往是由于相关技术企业无法在产品、服务或者营销内容的实效性方面自圆其说。在今天的文章中，我们将一同纵观最近一段时间来炒作缠身的技术项目。

从当初的牛顿与苹果、索尼Betamax盒式录像机到如今的推送技术、Web电视以及电动汽车……这些事物之间到底有何共性？一言以蔽之，它们最终成功冲破了炒作迷雾、向全世界证明了自身的真正实力。它们并不孤单，还有更多新兴项目有望证明实力。

事实上，今天的“下一波浪潮”很可能成为明天的炒作话题——或者永无出头之日，或者真正爆发出强劲潜能。

那么就当下的情况看，有哪些技术未能成功摆脱炒作的影响？我们采访了数十位IT专业人士、营销专家以及企业管理者，希望能找到正确的答案。下面我们就一同来看近年来炒作风头最盛的十二项新兴技术。

1. 大数据。“大数据已经在两个方面被过度拔高，”Scribe软件公司产品管理副总裁Betsy Bilhorn表示，这是一家数据整合与数据迁移软件供应商。“首先，很多企业试图将全部客户数据片段拼凑起来以创造出‘全面性视角’，从而‘将梦想变成现实’，”她指出。

“数据相关范围的过度膨胀导致其内容极为松散——企业需要的是全面且具备可操作性的客户数据，但过度深入了解每一个细节反而会埋没掉真正的重要信息，”Bilhorn指出。

“其次，”Bilhorn补充称，“大数据这一术语已经被用于形容各类相关数据。根据交流对象的不同，大数据可以指代技术基础设施（Hadoop）、非结构化数据/SQL或者将来自多个来源的数据加以结合。由于缺乏标准化定义，大数据概念将变得毫无意义。这像是把一切都称为‘云计算’——既模糊又可笑。”

“尽管大数据已经成为几乎每家技术企业天天挂在嘴边的话题，但其实际采纳比例还无法达到与炒作程度相符的临界点，”Virtusa公司全球技术解决方案部门高级副总裁Frank Palermo表示，这是一家全球性IT服务企业。

“客户对大数据实际效果的理解还非常模糊，而在大数据的影响全面降临之前这一状况很难得到显著扭转，”Palermo解释道。事实上，“根据Gartner公司的调查，大数据正处于‘期望值极度膨胀阶段’，而企业只会在某种技术进入炒作周期中的‘稳定生产阶段’时才会真正着手尝试，”他总结称。

2. QR码。“大家都在讨论这项技术，但大部分人还不清楚QR码到底是什么。而那些了解其定义的人往往根本不会尝试进行扫描，”电子商务顾问Ron Rule表示。“QR码确实是一项效果显著的便利因素，前提是我们已经在印刷品上看中了某种产品并有意愿通过网络进行购买。但宣传炒作把QR码形容成电子优惠券的便捷版本……这根本不是该技术的重点，”他指出。

QR码的问题在于，“大部分客户根本不会第一时间尝试使用它，因为他们完全不了解该如何开始，”GovernmentAuctions.org公司联合创始人兼CEO Ian Aronovich指出。“尽管很多企业都开始将QR码引入其广告及营销活动，但这并不会像社交媒体以及传统电子邮件营销等常规策略那样带来明显而且可靠的收效。”

3. 3D打印。“如果大家听信了各制造商们的宣传，那么3D打印机似乎已经成为生活中必不可少的实用性工具，”NetCablesPlus公司CEO JR Rodrigues表示。“然而我从用户群体中得到的实际反馈则是，由于喷嘴堵塞、印刷不均匀、软件故障以及整个打印周期时间过长等问题的存在，大家根本忍受不了这种新兴技术的过度青涩。顺带一提，打印数据文件的生成速度也是相当相当缓慢。”

“我们听到大量宣称3D打印技术将改变世界的新闻——最终这也确实可能成为现实，”解决方案集成商Kikata公司CEO Todd Emerson指出。“学校及企业对3D打印机的需求确实很旺，消费级3D打印机产品也开始逐步进入市场。然而能够创建3D模型的打印工具目前仍然算是比较复杂的技术，”他表示。“结果就是，这些3D打印机往往在被买下之后就很快走进仓库里吃灰。也许以Leap为代表的其它一些热门技术有可能解决3D渲染难题，成功让最年轻、技术积累最薄弱的用户学会如何创建3D模型，但至少就目前来看这还只是种美好的愿望。”

4. 游戏化。“游戏化如今确实存在过度炒作，”移动业务应用开发商Problemio公司创始人Alex Genadinik表示。“它几乎无法为产品增加大量价值，且在可操作性方面也进展缓慢。”

5. 指纹识别。“指纹扫描仪的介入仍然无法防止我们的iPhone被小偷盗走，”网络安全企业HBGary公司威胁情报部门主管Matthew Standart指出。“它确实能在一定程度上打消窃贼们的犯罪念头，但效果非常有限——会有多少坏人愿意在下手之前检查被害者移动设备上的身份验证机制？”他反问道。“而且只要内容未经加密，窃贼仍然有能力访问到设备中的数据。”

6. Siri。“走在大街上的时候，大家见过有人当众使用Siri的吗？这种技术真比手动方式更便捷吗？我们提出问题后，她能正确理解并做出反馈的比率有多少？”Automic公司产品管理与营销高级主管Kerry Lebel提出一连串问题——这是一家专门提供业务自动化平台的企业。

“Siri几乎无法处理超过一个词的企业名称，会为了等待更好的应答结果而卡住不动，而且大多数情况下的回应都只是‘对不起，我不明白，’”他抱怨道。虽然人们在第一印象中会认为Siri的出现既新颖又“酷劲十足”，但其令人失望的实际效果会迅速让用户变得沮丧--他们往往重新尝试手动输入以取得更快、更好的处理结果。

7. 近场通信（简称NFC）。“人们普遍认为近场通信将彻底改变消费活动的支付方式，而且不同用户似乎在一夜之间就建立起了新的沟通桥梁，”品牌体验机构Phenomblue技术主管Ryan Phelan表示。“但实际情况是，虽然目前支持NFC的设备已经不少，但它在真实生活中的表现还非常有限，”他指出。

“很多人希望NFC技术能够最终让移动钱包取代传统信用卡，”Palermo告诉我们。“然而由于缺乏通行标准、读取终端机数量严重不足以及iPhone等主流设备并不支持NFC，这项技术目前

还没有走上正轨。”

“NFC本应该成为移动支付机制的未来方向，”SAP公司移动商务高级副总裁Matthew Talbot表示。“不过在今年六月，Gartner公司将NFC的预期支付量下调了40%，这是因为谷歌钱包与ISIS都没能带来理想的市场表现。NFC并未彻底消亡，但作为一项得到极高关注的技术，它到今年年底只会占到整体交易支付款项中的2%、且到2017年这一比例预计也只能达到5%——这实在令人感到失望。”

8. 数字化/移动钱包。“移动钱包，也被称为数字化钱包，在过去两年中一直传闻缠身、关注度极高，然而却由于缺乏关键性发展拐点而始终未能真正起飞-其中涉及的正是现实版的‘先有鸡还是先有蛋’难题，”CardPaymentOptions.com公司CEO兼总裁Phillip Parker指出。

“商家必须投资新设备来支持这项技术，但除非客户提出强烈愿望、否则没人愿意主动掏这笔冤枉钱，”Parker解释道。“客户这边也一直在犹豫，因为支持这项技术的商家一直不多。除此之外，服务的碎片化特性也给商家及客户提出了兼容性难题。总之这些目前来看还太过复杂。”

9. 增强现实（简称AR）。“近来最具爆炸性的增强现实技术成果要数谷歌公司高调推出的谷歌眼镜产品，”Phelan表示。“不过这款设备的操作相当繁琐，而且功能也太过有限，”他指出。问题在于：“增强现实还没有彻底摆脱过去、步入全新发展阶段，目前没有一套真正能为消费者带来实际意义的机制促使我们购买。”

10. 穿戴式技术。“在消费领域，我们经历了大量关于穿戴式技术的炒作与铺天盖地的宣传——‘智能’手表与谷歌眼镜就是最典型的例子”移动安全解决方案供应商Good Technology公司首席技术官Nicko van Someren表示。

“从企业的角度来看，从此类设备当中受益的前景还一片朦胧——因为目前的方案在用户交互效果方面还太过有限，”他指出。“除此之外，用户验证机制在这样的用户界面当中也很难实现；但由于此类设备很容易被盗走，因此它所带来的潜在风险就非常严重……这最终会对企业数据造成安全威胁。”

11. 实时营销。“大部分正在或者尝试采用实时营销方案的品牌都出于被迫或者非自然选择——而其实际效果最终还是要归功于社交网站，”数字化机构Digitaria副总裁兼宣传部长Tom Siebert表示。

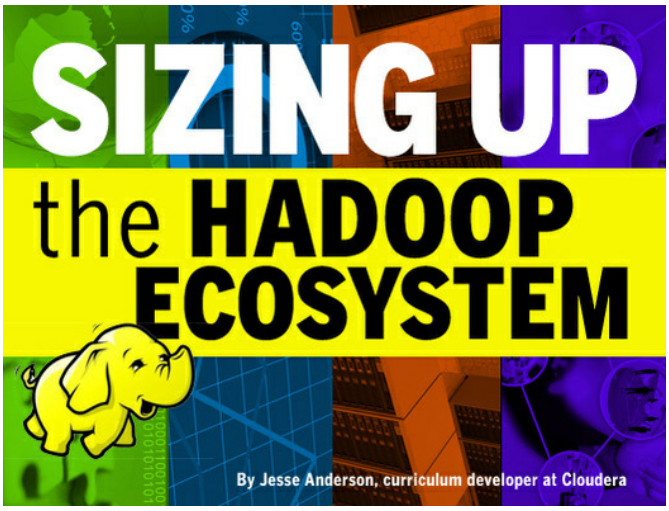
“此外，还没有人真正从‘实时营销’机制当中获得实惠；我们必须花上几个月甚至几年时间来拓展关注者乃至粉丝清单，”Siebert告诉我们。“到那个时候，我们的‘实时’营销机会早就消失得无影无踪了——这其实是一种需要花费数年时间进行开发的长周期产品。”

12. 团购网站。“就在不久之前，Groupon拒绝了谷歌公司的高额收购请求，转而准备自己的首次公开募股，”Compuware专业服务公司首席数字战略官Matthew David指出。

“时至今日，Groupon、LivingSocial以及其它一些团购网站的未来已经黯淡下来。除了与这些网站合作有可能给企业自身带来损害之外，团购券技术本身非常容易复制及改进，”David解释道。“团购风潮之所以未能长足发展，主要是因为它没能找到真正适合自己的市场定位——团购网站应该与社交网络类似，应该是另大实体企业的组成部分。■

Hadoop生态系统全面盘点

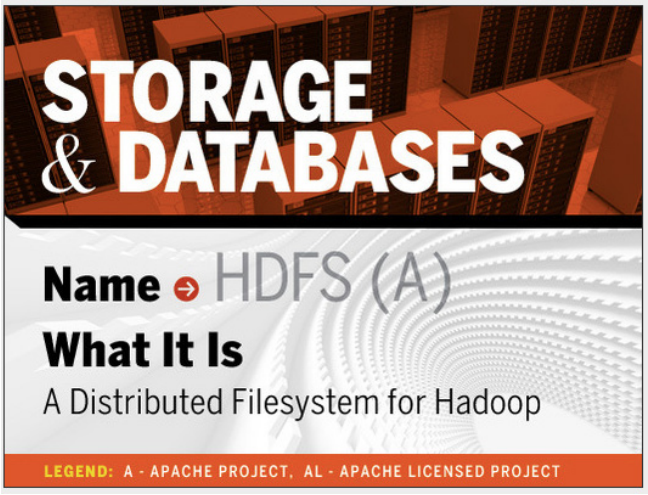
□ 核子可乐译/译



Hadoop拥有一个庞大而且充满活力的开发者社区，但Hadoop生态系统当中的很多项目在名称与功能的匹配性方面出入很大，这直接导致很多朋友难以分清各个项目的作用以及使用目的。

作为Hadoop的构建核心，HDFS与MapReduce是无法回避的两大要素。不过开发者及企业做出的大量贡献让Hadoop成为一套更为复杂的平台。生态系统中的一部分项目属于Apache基金会项目（在下图中以‘A’作为标注），另一些则属于具备Apache许可、但却由企业负责运作（在下图中以‘AL’作为标注）的项目。

在今天的文章中，我们将对Hadoop的发展历程做一番回顾，看看如今其阵容之中包含着哪些“猛将”。

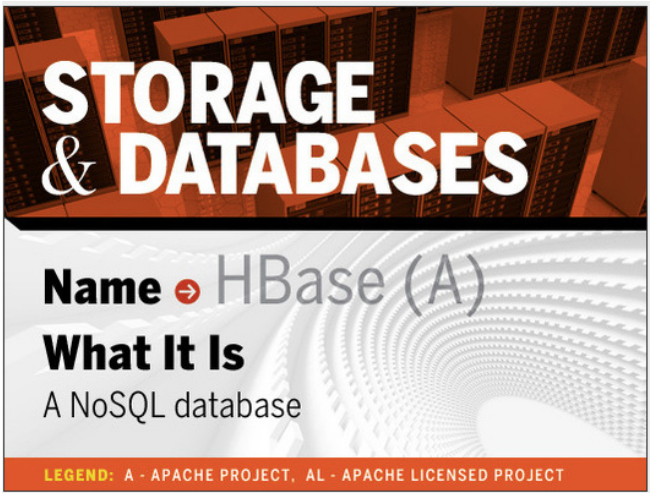


能做什么：作为Hadoop的文件系统或者存储机制。

有何帮助：创建一套重复性、可容错且可扩展的文件系统，旨在处理大型文件。利用数据位置提高MapReduce任务的数据输入性能。

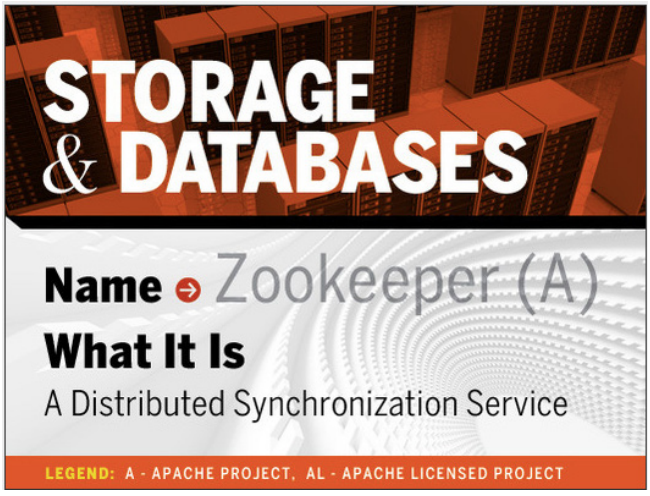
能做什么: 一款高度可扩展的数据库。

有何帮助: 允许我们以线性方式对数据库进行扩展。提供可进行调节的数据一致性级别控制。



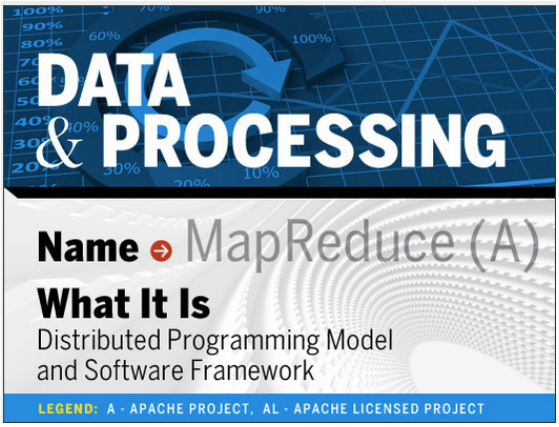
能做什么: 利用HDFS创建高度可扩展性数据库。

有何帮助: 实现高度可扩展特性与随机访问。利用HDFS保障严格的数据一致性。



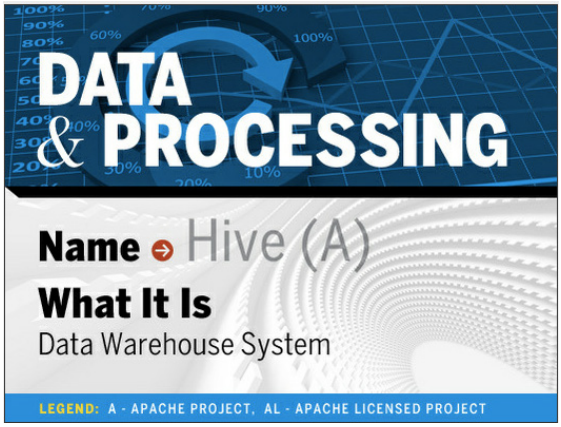
能做什么: 帮助分布式节点之间的数据实现同步 。

有何帮助: 对集群中所有节点之间的一致性分布式小规模数据进行维护。



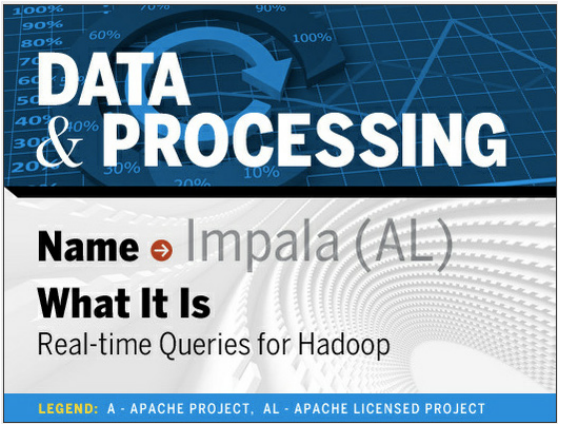
能做什么: 将一项工作拆分为多个任务并同时加以实施。

有何帮助: 框架会对分布式系统当中的疑难组件进行抽象化处理，允许系统同时处理大量数据。



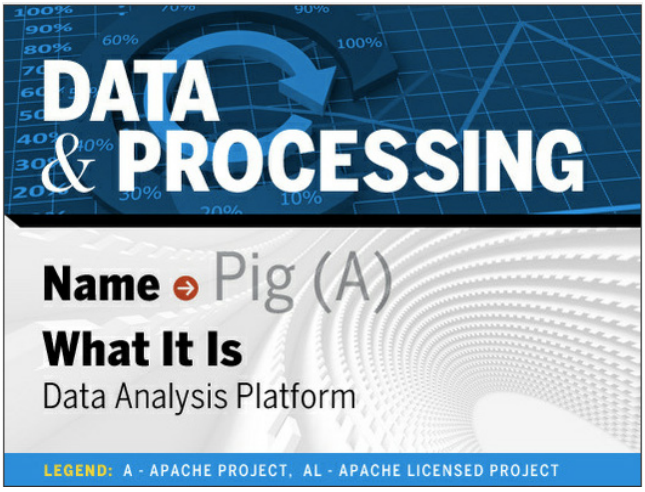
能做什么: 允许用户通过查询语言来处理数据。

有何帮助: 帮助SQL程序员通过创建类SQL查询以使用MapReduce。



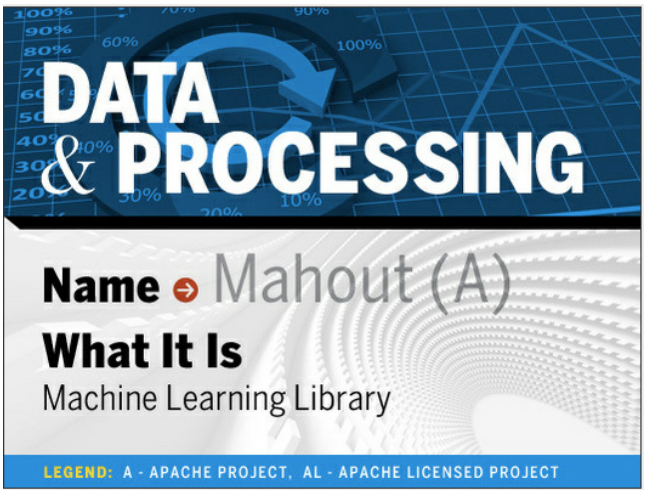
能做什么：在大规模数据当中实现低延迟查询。

有何帮助：帮助SQL程序员通过创建类SQL查询以提高大数据访问速度。



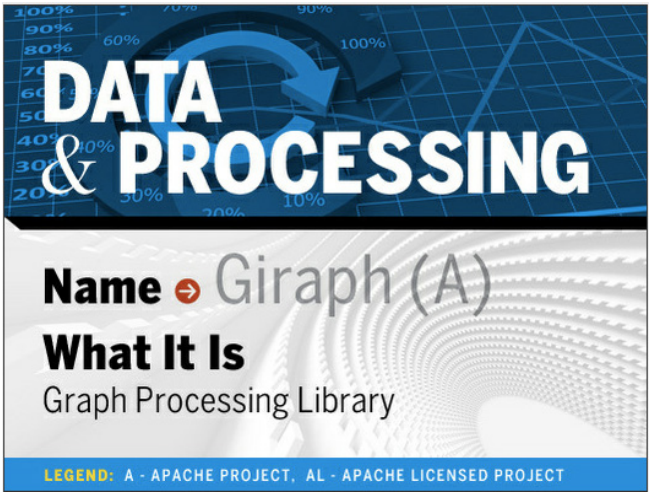
能做什么：利用数据流或者脚本类语言进行数据处理。

有何帮助：帮助程序员利用数据流语言发挥MapReduce功能。



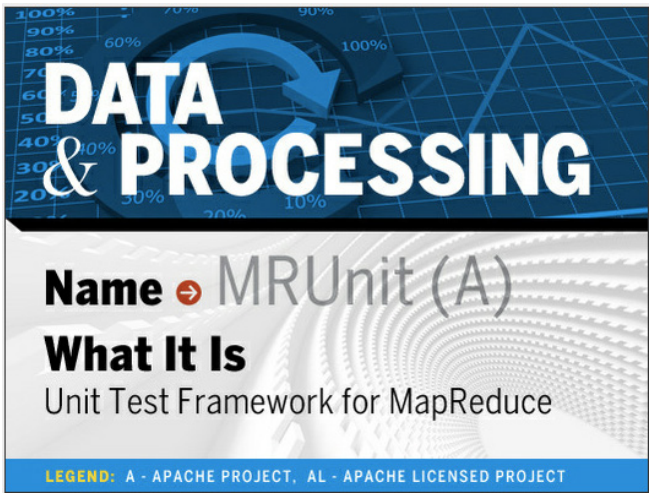
能做什么：利用预先编写的库在MapReduce中运行机器学习算法。

有何帮助：允许用户通过库创建MapReduce建议与集群。利用现有代码加快开发速度。



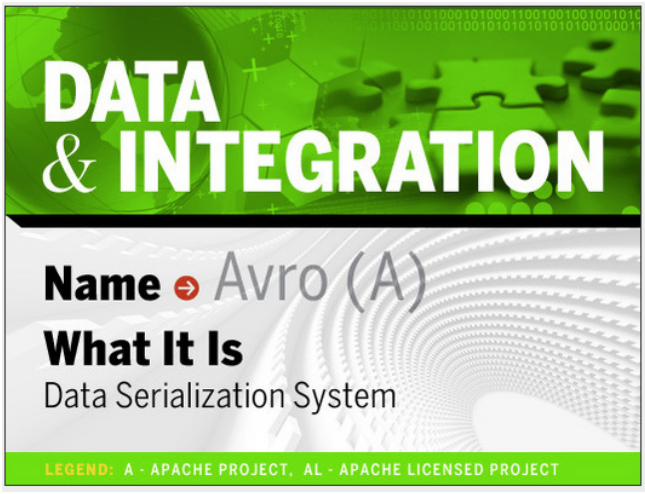
能做什么：利用预先编写的库在MapReduce中运行图形算法。

有何帮助：用户不必再为了使用MapReduce而重新编写图形算法。利用现有代码加快开发速度。



能做什么：运行测试以验证MapReduce工作功能的正确性。

有何帮助：运行程序测试以验证MapReduce程序运作的正确性。提供对象，允许用户模拟输入流程以验证执行结果。

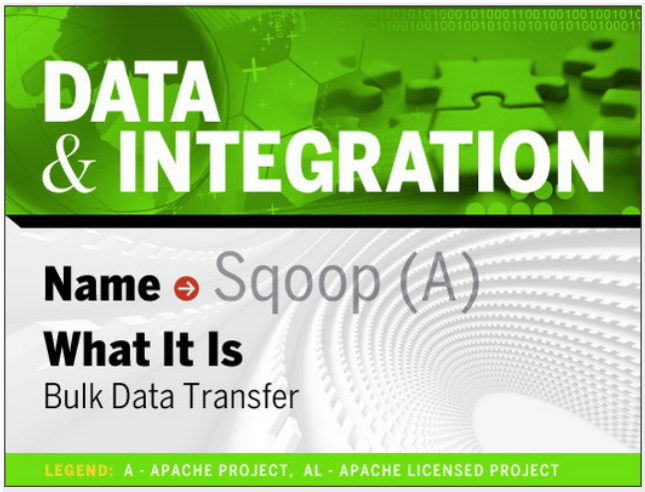


能做什么：提供简便方式以实现MapReduce工作数据的输入与输出。

有何帮助：创建域对象以存储数据。简化MapReduce工作数据的序列化与反序列化。

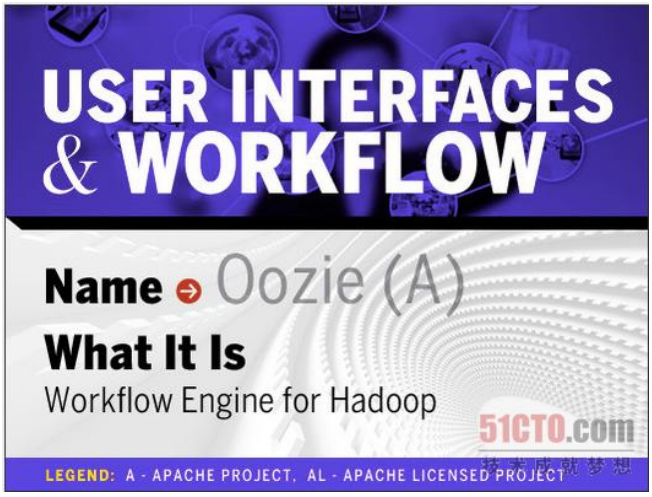
能做什么：以可扩展方式处理大量日志数据。

有何帮助：将大量日志数据转移到HDFS当中。由于Flume拥有的扩展性，因此能够处理大量传入数据。



能做什么：在关系型数据库与Hadoop之间实现数据移动。

有何帮助：允许数据从关系型数据库转移到Hadoop当中，以备后期处理。将数据输出结果从MapReduce工作中提取出来并移回关系型数据库。



能做什么：允许用户通过网络浏览器与Hadoop集群进行交互。

有何帮助：用户能够更轻松地与Hadoop集群实现交互。细化权限设定帮助管理员对用户进行配置。■

将彻底改变我们生活的十大现实世界大数据部署方案

如今全球数据量正迅猛增长，每十八个月总量就会翻上一倍。就在不知不觉之中，我们身边的现实世界已经开始转向由大数据驱动的新时代。在今天的文章中，我将带大家一同回顾十大大数据部署实例。

关于大数据话题的炒作与争论似乎永无停歇，但全球数据量迅猛增长、每十八个月总量翻上一倍的客观现实却没人能够否认。对这些新增数据的利用已经延伸到我们生活中的几乎各个方面，只是有些相对直观、有些却在悄然发生。今天我们就来一同回顾那些不为人知却实际存在的十大大数据部署案例。

Netflix

Netflix已经成为美国国内规模最大的商业视频流供应商——目前拥有2900万视频流客户。这家公司同时也成为吸收新增数据的海绵——用户在看什么、喜欢在什么时段观看、在哪里观看以及使用哪些设备观看，爆增的信息量成为Netflix手中的宝贵资产。他们甚至掌握着用户在哪视频的哪个时间点后退、快进或者暂停，乃至看到哪里直接将视频关掉等信息。现在Netflix公司开始推出自己的原创节目，而节目制作的依据正是刚刚提到的这些数据。他们利用手中的数据说服BBC重新翻拍了电视连结剧《纸牌屋》；而且将演员Kevin Spacey与导演David Fincher的粉丝与原剧集支持者的粉丝进行关联，最终让这二位加盟新剧的拍摄。

Ancestry.com

Ancestry.com帮助人们将自己与家庭史结合起来并创建独一无二的树状家谱。从表面上看，这个主意似乎没什么技术含量，但为了实现这项功能、网站需要维护超过110亿条记录与高达4PB的数据量——其中包括历史记录、出生记录、死亡记录、战争与移动记录甚至年鉴等——其中不少往往采取手写格式。它利用高级内容处理技术对全部相关信息加以索引，从而保证数据

的可搜索性。Ancestry.com还引入额外的DNA处理结果以生成新型数据流，从而帮助客户更准确地建立血缘关系。通过对唾液进行采样，网站方面能够对客户的DNS进行排序并将结果与数据库中的其它客户加以匹配——例如找到多年没有联系的表亲。

西奈山医疗中心

西奈山医疗中心是美国历史最悠久、规模最大的教学型医院之一，其在医学教育与生物医学研究方面的地位非常突出。目前中心方面正利用来自大数据新兴企业Ayasdi公司的技术对整个大肠杆菌基因组序列进行分析，其中包括超过100万个DNA变异，旨在努力理解某些菌株如何在与抗生素的共处中获得抗药性。细菌的抗药性影响着全球各地数以百万计的病人。Ayasdi的技术为数学研究、拓扑数据分析（简称TDA）开辟了一片新天地，有助于人们更深刻地理解数据形态。

加利福尼亚州ISO

加州独立系统运营商（简称ISO）管理着全加州地区超过八成电网中的供电走向，每年提供的电力达到2.89亿千万时、惠及3500万民众，供电线路的总长度超过25000英里。他们利用Space-Time Insight公司的软件实现情景智能化机制，从而将来自多个来源的大规模数据进行关联与分析——其中包括天气状况、传感器数据以及计量设备测绘结果等——并以可视化形式帮助用户查看并理解如何对可再生能源进行优化、实现整个电网的电力供需平衡并快速应对潜在危机。

Hydro One网络

Hydro One公司负责为全安大略省的家庭及企业提供电力。该公司拥有并经营安大略省内总长达29000公里的高压输电网络以及总长达123000公里、直接面向130万用户的低压配电系统。Hydro One使用的是由Space-Time Insight提供的地理空间与可视化分析软件，旨在改进当前输电与配电资产的健康性与可靠性。这套系统能帮助资产管理者及时获取相关情报，包括资产性能随时间推移而发生的变化、资产更换战略以及资产维护需求等。该方案还能将数据与

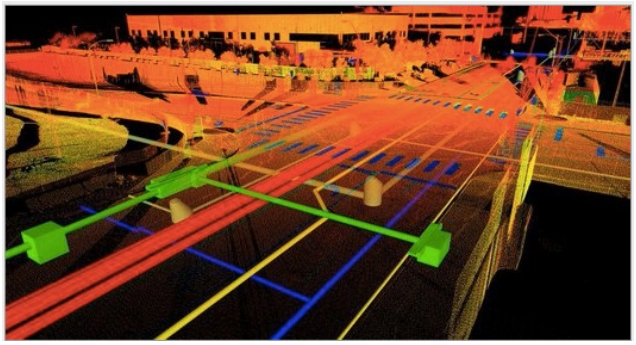
其它多种不同系统的功能结合起来，包括SAP ECC、SAP BW、GIS系统以及实时数据等，从而帮助Hydro One对自身拥有的资产具备宏观掌控能力。

俄勒冈健康与科学大学

俄勒冈健康与科学大家（简称OHSU）是位于美国俄勒冈州的一所公立大学，下辖两所医院、一座一级创伤恢复中心和一家儿童医院。校方将Stanley Black与Decker Disivion Stanley Healthcare提供的MobileView软件与Tableau软件的数据虚拟化技术结合起来，旨在追踪院内4000个注液泵的实时位置与工作状态，从而掌握注入到患者循环系统当中的液体、药物或者营养物质——事实上，这项工作如果完全依靠手动方式执行、其可靠程度将大打折扣。该技术还允许校方对历史及当前资产数量进行分析，进而更好地规划未来数量水平、提高库存物资的分配与利用效率。

拉斯维加斯市

由于记录太过古老、信息不够准确，大部分城市中的公共事业机构都不了解埋在地下的资产处于何种状况——因此居民往往会由于某条供电线被意外切断或者某条供水管线老化爆裂而受到影响。为了避免这些难题，拉斯维加斯市采取智能数据方式开发出一套实时公共事业网络模型。VTN咨询公司帮助市政当局通过各种渠道汇总数据，并利用Autodesk技术创建出实时3D模型。这套模型中包含着地上与地下的所有公共设施，目前已经被用于监测城市地下设施的具体位置以及运转状况。



迈阿密市属戴德县

佛罗里达州迈阿密市属戴德县正积极响应IBM提出的智能化城市倡议，希望将35个区域自治单位与迈阿密市聚拢起来，努力帮助政府领导做出更为明知的管理决策——包括充分利用水资源、减少交通拥堵以及改善公众安全等。IBM通过云计算环境下的深层分析为该县带来一套情报仪表板，从而帮助各机关与部门彼此协作并实现可视化管理。举例来说，戴德县县公园部门今年预计将通过识别并修复因锈蚀而漏水的浇灌管道节省100万美元经费。

澳大利亚网球协会

在一年的大多数时段内，作为澳大利亚网球公开赛的经营方，澳大利亚网球协会的运作状态与普通的小型企业没什么差别。然而一旦为期两周的澳网公开赛开始进行，协会瞬间就成了一家规模庞大、对数据极度渴求的大型企业——他们需要不间断地访问准确内容、数据以及统计结果，从而进行分析并做出决策。澳大利亚网球协会采用IBM的实时数据分析软件来检查赛程进行状态、运动员人气、历史数据记录以及社交媒体上球迷们对比赛网站提出的数据需求。根据实际需求，这项技术能够为分析工作分配必要的计算资源。

DPR Construction

DPR Construction公司是加州大学旧金山分校斥资15亿美元在米慎湾兴建的医学中心的承包商，这也是第一座建造时间超过十年的医学中心。DPR利用来自Autodesk公司的3D技术帮助手下的设计师们收集空气流量、建筑物朝向、楼体间距、环境永续性以及建筑性能等数据，并将结果导入到一套单独的虚拟模型当中。通过这种方式，建筑师、设计师以及旗工队伍能够以可视化方式掌握遍布整个运作环境下的数亿个数据标记。■

专题推荐

七步走：AngularJS从菜鸟到专家

编者按

对于众多企业来讲，去O进入”My”世界还是行不通的，当然这是一个趋势，并不是一个最佳选择。与互联网目前的形势来看，传统IT架构还是比较重要，跟规模与需求无关。

去IOE化浅谈：能否去“O”进入“My”世界

在去年，阿里巴巴集团将正式公布技术团队合并的事情后，以此同时，也实现了用开源的MySQL数据库替换Oracle数据库。2013年6月20日，阿里高调公布其对为支付宝用户服务了5年的最后一台小型机下线。在阿里巴巴集团首席架构师王坚的主导下，采用PC-Server承载MySQL数据库，支撑大并发大数据量的核心业务系统。

在全新的IT架构下，淘宝的业务变得更加开放、灵活、高效，并在中国的互联网行业产生了很大的影响，欲效仿者甚众。当然了，如果企业没有总结和沉淀积累，绝对不会盲目去实践。

下面笔者浅谈对去Oracle化进入MySQL世界的一些观点，欢迎拍砖。

技术

笔者认为，在谈MySQL的同时我们要知道开源软件跟商业产品最重要的一个区别就是，开源软件其实只是一个基本上不成熟的框架，后期需要企业与相关业务去磨合，如果碰到缺陷更没有厂商对技术的支持。

没有一个庞大的技术后盾，冒然使用开源软件结合自己的业务，这是一件很危险的事情。阿里其实应该属于去IOE最彻底的公司。在初期，阿里和淘宝曾经多次尝试从Oracle数据库迁移到MySQL，但是都失败了，最根本的原因就是懂MySQL DBA技术的人才极少。

了解“棱镜门”事件的网友应该知道，目前国内的IT技术积累远远不够。要让企业真正的用MySQL完全替换Oracle是一件不现实的事情，阿里技术保障部DBA负责人周宝方曾说过，“去IOE”技术门槛很高，其他企业根本无法复制。

成本

去Oracle这不是一个成本的问题，但你又不得不去考虑它的成本。首先，这是一个技术的替换过程，其次必然会考虑更加省钱的数据库软硬件解决方案，最后就是时间成本问题。很多人在谈阿里的成功，却忽视它背后花了2年多的时间用重金打造的一个“成长”过程换来如今的低基础架构成本与高工作效率。

距11g后，Oracle在前段时间推进12c，具有

的高性能、轻量级的数据库性能监控、解决了11g中EM（控制台）的庞大臃肿、易管理等优势，总成本无疑是降低的。再者，有非常多的技术细节以及12c许可方式的改变等等。对于用户来说，负担一定是减少了，因为系统资源的占用大大减少。如果在按照CPU计费的标准下，用户的许可的费用也是大大减少。此外笔者觉得如果换成MySQL后，虽然解决了软件在成本方面的问题，但是在使用的过程中它的升级和维护成本往往会牵扯到更多的人力和财力。

如果你认为去Oracle替换MySQL是为了降低成本，有人就笑了。

▲人力成本的体现

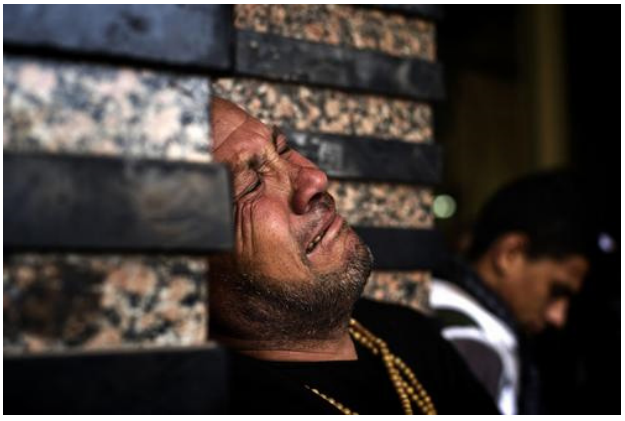
客户

说到客户，笔者认为大部分客户都是比较青睐Oracle的集成系统，再者，Oracle在收购SUN后MySQL也属于它的产品，就有人担心MySQL的前途。

像淘宝这样以及其他电商企业客户群体大部分都是消费者，客户的顾虑很简单，在替换MySQL后没有谁会考虑你的规模，处理能力等各方面，他们只需顺利完成购物交易行为，就肯定你的选择是正确的。

如果是传统企业，以金融行业为例，目前国内就难以找到让客户满意又能替代Oracle的产品。原因其实很简单，在传统企业IT基础架构不是很完善，缺乏最佳实践，经常面临捉襟见肘的局面。如果冒然替换，不但起不到成本降低的要求，低效的工作会让自己的用户无法满足。另外一个问题，Oracle集成体系在传统行业已经根深蒂固，几乎渗透到金融业，运输业，电信业，连锁业等等，客户也是在这个过程中形成了一种依赖。

在这个要处理半结构化和非结构化的大数据时代，笔者认为推广Oracle恰是一个最佳的选择。



▲你认为再好，在客户的眼里都是渣

总结：

综合上述，对于众多企业来讲，用去“O”进入“My“世界还是行不通的，当然这是一个趋势，并不是一个最佳选择。与互联网目前的形势来看，传统IT架构还是比较重要，跟规模与需求无关。想进入“My”世界，且要师夷长技以制夷。■

HTML5 Indexed DB入门导学【2】

■ 这里要说的都是革新，说这些的目的就是要保持关注最新技术。如果你是一个程序员，想要探寻未来技术，那这篇文章就是你的必读之选。我们这里列出了10种编程语言，10种将会改变IT世界工作方式的编程语言。这些语言已经在开始改变IT界的景象。

在本系列教程的第一篇http://developer.51cto.com/art/201310/412224.htm 中,我们向大家介绍了IndexedDB的基本知识,在本篇教程中,将继续向大家介绍如何使用IndexedDB进行CRUD(增删改查)的功能,其中将特别将讲解IndexedDB的更新和删除功能,并且将会演示一个实际的例子,这个例子在第三篇教程中将会用到。

更新记录

首先看下如何在IndexedDB中更新记录。如果记得上一篇教程的话，增加数据是很简单的，如下：

```
1. //Define a person
2. var person = {
3.     name:name,
4.     email:email,
5.     created:new Date()
6. }
7. //Perform the add
8. var request = store.add(person);
```

对于更新操作，如果已经在对象中已经定义了id作为key，则只需要使用put方法代替add方法即可，代码如下：

```
1. var person = {
2.     name:name,
3.     email:email,
4.     created:new Date(),
5.     id:someId
6. }
7.
8. //Perform the update
9. var request = store.put(person);
```

和增加的方法一样，可以指定各类回调方法对结果进行异步处理。

删除记录

删除记录的方法是通过delete这个API去实现，下面是例子：

```
1. var t = db.transaction([“people”],
   “readwrite”);
2. var request = t.objectStore(“people”).
   delete(thisId);
```

同样，可以指定各类回调方法对结果进行异步处理。

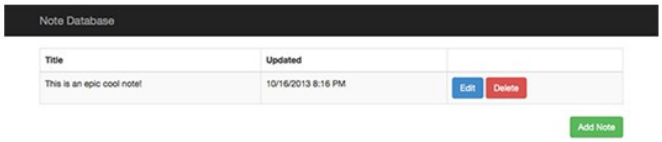
在学会了如何使用IndexedDB进行CRUD后，下面我们以一个实际的例子来设计一个简单的记事本应用。

记事本应用

下面我们来学习如何使用学到的IndexedDB知识，设计一个简单的记事本应用。在应用启动的时候，先初始化IndexedDB数据库，并且展示一个空的表格，能让用户增加空的记录。界面如下图：

当先“Add Note”按钮后，出现的界面如下图：

可以在这里输入要记录的事情，然后点Save键保存。保存后的列表界面如下图：



同时用户可以再次编辑或删除记事的内容。下面我们首先开始设计第一个页面，注意我们使用的是bootstrap框架。

```
1. <!DOCTYPE html>
2. <html lang=” en” >
3.  <head>
4.   <meta charset=” utf-8” >
5.   <meta name=” viewport”
   content=” width=device-width, initial-
   scale=1.0” >
6.
7.   <title>Note Database</title>
8.
9.   <link href=” bootstrap/css/bootstrap.
   css” rel=” stylesheet” >
10.    <link href=” css/app.css”
   rel=” stylesheet” >
11.
12.    </head>
13.
14.    <body>
15.
16.     <div class=” navbar navbar-inverse
   navbar-fixed-top” >
17.      <div class=” container” >
18.       <div class=” navbar-header” >
19.        <a class=” navbar-brand”
   href=” #” >Note Database</a>
20.       </div>
21.      </div>
22.     </div>
23.
24.     <div class=” container” >
25.
26.      <div id=” noteList” ></div>
27.      <div class=” pull-right” ><button
   id=” addNoteButton” class=” btn btn-
   success” >Add Note</button></div>
28.      <div id=” noteDetail” ></div>
```

```
29.
30.     <div id=”noteForm”>
31.     <h2>Edit Note</h2>
32.     <form role=”form” class=”form-
  horizontal”>
33.     <input type=”hidden” id=”key”>
34.       <div class=”form-group”>
35.         <label for=”title” class=”col-lg-2
  control-label”>Title</label>
36.         <div class=”col-lg-10”>
37.         <input type=”text” id=”title”
  required class=”form-control”>
38.           </div>
39.         </div>
40.       <div class=”form-group”>
41.       <label for=”body” class=”col-lg-2
  control-label”>Body</label>
42.       <div class=”col-lg-10”>
43.       <textarea id=”body” required
  class=”form-control”></textarea>
44.         </div>
45.       </div>
46.     <div class=”form-group”>
47.     <div class=”col-lg-offset-2 col-lg-10”>
48.     <button id=”saveNoteButton”
  class=”btn btn-default”>Save Note</button>
49.     </div>
50.     </div>
51.     </form>
52.     </div>
53.       </div>
54.       <script src=”js/jquery-2.0.0.min.
  js”></script>
55.       <script src=”bootstrap/js/bootstrap.
  min.js”></script>
56.       <script src=”js/app.js”></script>
57.     </body>
58.   </html>
```

接下来是编写核心的js文件，首先是一个处理日期的工具类，比较简单：

```
1. /* global console,$,document,window,alert */
2. var db;
3.
4. function dtFormat(input) {
5.   if(!input) return “”;
6.   var res = (input.getMonth()+1) + “/”
  + input.getDate() + “/” + input.getFullYear()
  + “ “;
7.   var hour = input.getHours();
8.   var ampm = “AM”;
9.   if(hour === 12) ampm = “PM”;
10.     if(hour > 12){
11.       hour-=12;
12.       ampm = “PM”;
13.     }
14.     var minute = input.getMinutes()+1;
15.     if(minute < 10) minute = “0” +
  minute;
16.     res += hour + “:” + minute + “ “ +
  ampm;
```

```
17.         return res;
18.     }
```

接下来的代码中，要首先检查浏览器是否支持IndexedDB，如果支持IndexedDB，则打开数据库，代码如下：

```
1. $(document).ready(function() {
2.
3.     if(!(“indexedDB” in window)) {
4.         alert(“IndexedDB support required for this demo!”);
5.         return;
6.     }
7.
8.     var $noteDetail = $(“#noteDetail”);
9.     var $noteForm = $(“#noteForm”);
10.
11.         var openRequest = window.indexedDB.open(“nettuts_notes_1”,1);
12.
13.         openRequest.onerror = function(e) {
14.             console.log(“Error opening db”);
15.             console.dir(e);
16.         };
17.
18.         openRequest.onupgradeneeded = function(e) {
19.
20.             var thisDb = e.target.result;
21.             var objectStore;
22.
23.             //Create Note OS
24.             if(!thisDb.objectStoreNames.contains(“note”)) {
25.                 console.log(“I need to make the note objectstore”);
26.                 objectStore = thisDb.createObjectStore(“note”, { keyPath: “id”, autoIncrement:true });
27.             }
28.
29.         };
30.
31.         openRequest.onsuccess = function(e) {
32.             db = e.target.result;
33.
34.             db.onerror = function(event) {
35.               // Generic error handler for all errors targeted at this database’s
36.               // requests!
37.               alert(“Database error: “ + event.target.errorCode);
38.               console.dir(event.target);
39.             };
40.             displayNotes();
41.         };
```

在上面的代码中，首先判断用户的浏览器是否支持IndexedDB，如果支持的话，则打开数据库，在其中的onupgradeneeded事件中建立名为note的objectstore，并且注意这里要编写对应

的操作成功和失败的回调处理事件。一切准备就绪后，则可以调用displayNotes()方法显示所有的记事列表，其代码如下：

```
1. function displayNotes() {
2.
3.    var transaction = db.
   transaction([“note”], “readonly”);
4.    var content=”<table class=’ table
   table-bordered table-
   striped’ ><thead><tr><th>Title</
   th><th>Updated</th><th>&amp;nbsp;</td></
   thead><tbody>”;
5.
6.    transaction.oncomplete =
   function(event) {
7.        $(“#noteList”).html(content);
8.    };
9.
10.        var handleResult = function(event) {
11.          var cursor = event.target.result;
12.          if (cursor) {
13.            content += “<tr data-
   key=\” ” +cursor.key+” \” ><td
   class=\” notetitle\” >” +cursor.value.title+” </
   td>”;
14.            content +=
    “<td>” +dtFormat(cursor.value.updated)+” </
   td>”;
15.
16.            content += “<td><a class=\” btn
   btn-primary edit\” >Edit</a> <a class=\” btn
   btn-danger delete\” >Delete</a></td>”;
17.            content +=” </tr>”;
18.            cursor.continue();
19.          }
20.          else {
21.            content += “</tbody></table>”;
22.          }
23.        };
24.
25.        var objectStore = transaction.
   objectStore(“note”);
26.
27.        objectStore.openCursor().onsuccess =
   handleResult;
28.
29.    }
```

在本系列教程的第一篇中，已经讨论过如何循环读取IndexedDB中的数据（使用的是游标cursor，还记得么？），但在上面的这段代码中，我们在objectStore.openCursor()事件的onsuccess方法中，指定了其回调处理代码为handleResult，而在handleResult方法中，完成了对数据库中数据的循环读取列表。而在IndexedDB中允许直接在最顶层的事务中绑定onComplete事件，这就给我们带来了方便，比如例子中可以将比如大量的文本输出（这里的content）交给前端jQuery进行显示处理。

接下来看删除，编辑和增加记事的功能代码：

```
1. $(“#noteList”).on(“click”, “a.delete”,
   function(e) {
```

```
2.    var thisId = $(this).parent().parent().
   data( “key” );
3.  var t = db.transaction([ “note” ],
    “readwrite” );
4.    var request = t.objectStore( “note” ).
   delete(thisId);
5.    t.oncomplete = function(event) {
6.        displayNotes();
7.        $noteDetail.hide();
8.        $noteForm.hide();
9.    };
10.         return false;
11.     });
12.
13.     $( “#noteList” ).on( “click” , “a.edit” ,
   function(e) {
14.         var thisId = $(this).parent().parent().
   data( “key” );
15.
16.         var request = db.transaction([ “note” ],
    “readwrite” )
17.                         .objectStore( “note” )
18.                         .get(thisId);
19.         request.onsuccess = function(event) {
20.             var note = request.result;
21.             $( “#key” ).val(note.id);
22.             $( “#title” ).val(note.title);
23.             $( “#body” ).val(note.body);
24.             $noteDetail.hide();
25.             $noteForm.show();
26.         };
27.
28.         return false;
29.     });
30.
31.     $( “#noteList” ).on( “click” , “td” ,
   function() {
32.         var thisId = $(this).parent().
   data( “key” );
33.         var transaction = db.
   transaction([ “note” ]);
34.         var objectStore = transaction.
   objectStore( “note” );
35.         var request = objectStore.get(thisId);
36.
37.         request.onsuccess = function(event) {
38.             var note = request.result;
39.             $noteDetail.html( “<h2>” +note.
   title+” </h2><p>” +note.body+” </p>” ).
   show();
40.             $noteForm.hide();
41.         };
42.     });
43.
44.     $( “#addNoteButton” ).on( “click” ,
   function(e) {
45.         $( “#title” ).val( “” );
46.         $( “#body” ).val( “” );
47.         $( “#key” ).val( “” );
48.         $noteDetail.hide();
49.         $noteForm.show();
50.     });
```

只要用户读过本系列教程的第一篇，就应该知道上面的代码并不复杂，关键点是无论是编辑还是删除记事，都是必须先找出其id,这其实通过普通的DOM操作就可以实现了。接下来看保存记事的代码：

```
1. $(“#saveNoteButton”).on(“click”,function()
   {
2.
3.        var title = $(“#title”).val();
4.        var body = $(“#body”).val();
5.        var key = $(“#key”).val();
6.
7.        var t = db.transaction([“note”],
   “readwrite”);
8.
9.        if(key === “”) {
10.               t.objectStore(“note”)
11.                        .add({title:title,body:bo
   dy,updated:new Date()});
12.           } else {
13.               t.objectStore(“note”)
14.                        .put({title:title,body:bod
   y,updated:new Date(),id:Number(key)});
15.           }
16.
17.           t.oncomplete = function(event) {
18.               $(“#key”).val(“”);
19.               $(“#title”).val(“”);
20.               $(“#body”).val(“”);
21.               displayNotes();
22.               $noteForm.hide();
23.           };
24.
25.           return false;
26.       });
27.
28.   });
```

在上面的代码中，通过IF语句去判断KEY是否为空，如果是空的话则表示是新增加记录，否则就是在更新记录。

在本系列教程的下一讲，将重点介绍IndexedDB 的检索功能和数组方面的功能。本讲的代码可以在如下连接下载：https://github.com/tutsplus/working-with-indexeddb-part-two

■

开发人员指南：如何为未来汽车技术做好开发准备？

核子可乐/译

忘记《杰特森一家》里那如梦似幻的太空飞轮吧，现在我们该从开发人员的角度出发，认真思考创新方向：嵌入式系统、网络挑战、人机交互以及软件标准。

1954年福特公司公布的FX-Atmos概念车

其实我们很容易理解为什么有那么多未来技术作者会把汽车视为衡量技术发展水平的重要依据。除了日常代步之外，汽车既是我们生活中价值最大的消费品之一、同时也代表着最前沿的创新成果。时至今日，我们已经开始在汽车上看到众多新功能，其中包括机载传感器、高速无线通信、云服务、实时数据访问以及个性化人机交互方案。对于消费者来说这当然是个好消息，但对于设计师、工程师以及开发人员来说，便利的背后则蕴藏着巨大的技术挑战。

如果大家曾经针对多核心系统开发出任何一种软件产品，那么应该会对汽车行业带来的技术挑战更为熟悉。简单来说，多核心架构在更强大的同时却也更复杂……而这就是一切的起点。我们不妨想象一下，对于一套以安全性为最高诉求的系统当中，成功或者失败可能将在一瞬间决定。

在今年十月初始举办的Grace Hopper计算纪念大会上，四位来自福特公司的工程师分享了他们个人对推动整个行业所做出的贡献——这不仅是福特的财富、同时也是整个交通运输行业的一部分。听了他们的陈述，我多次发出“哦，太酷了！”的赞叹。相信大家在本文的引导下，也会与我一样对美好未来充满期待。

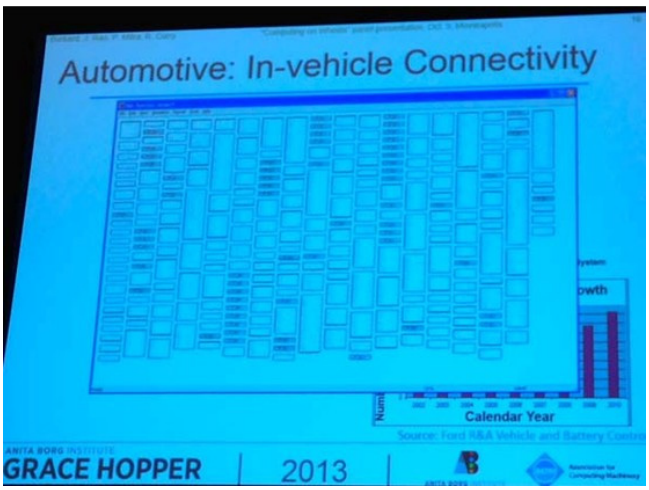
复杂性

拥有二十多年汽车行业从业经验的Dona Burkard解释称，汽车嵌入式系统的复杂性是个不容忽视的难题。顺带一提，她的专长领域在于动力总成内的嵌入式系统。

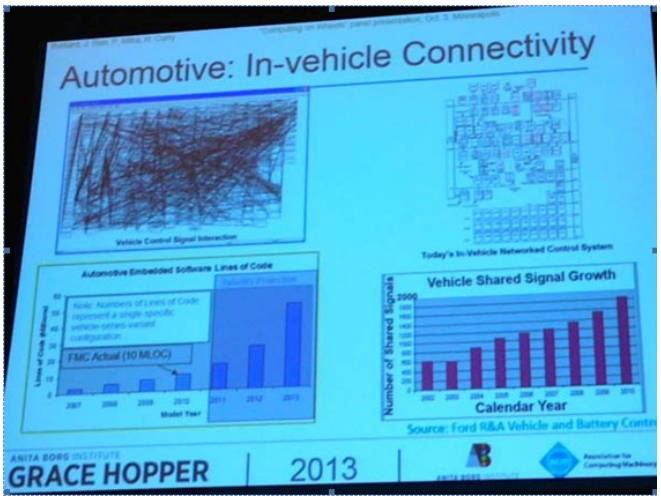
遥想2006年，当时一辆汽车上只存在13块嵌入式CPU（即ECU），她回忆称；时至今日，这一数量已经增加到65块。正如Burkard所说，多核心微控制器架构开始逐步入驻动力总成控制模块，这在持续优化发动机与变速箱功耗与散热状况的同时、也使二者的控制复杂性迎来大幅提升。由此带来的技术挑战在于：更困难的静态/动态分析；大量以复杂的手写方式开发出的传统应用在设计思路上并未

考虑并行系统；需要更为严格的软件标准等。

对于我们来说，一句“复杂性”似乎就能涵盖所有；但通过下图，大家才能真正了解现代汽车当中需要纳入考量的代码模块数量：



在再来想想我们该如何勾勒它们之间彼此作用的关系图。在下面的图片中，请大家关注左上方的图表（也就是那乱糟糟的一团），想想对其进行调试到底有多困难——更不用说重构多核硬件了。



行业标准的出台当然是件了事，但在任何知识领域，标准化进程往往行动迟缓——有时候甚至比客户们迫切要求使用的技术更滞后。对于交通运输行业，相关标准为ISO 26262与AutoSAR（即汽车开放系统架构），“它们被应用到众多安全相关系统当中，这些系统内包含一个或多个电气和/或电子系统（简称E/E）并已经被安装在交付生产的多种系列载客车辆当中。”

根据Burkard的说法，这带来的结果是福特需要提前数年与制造商及OEM厂商携手创建一套集成化平台。这通常代表着平台的建立甚至要比具体车型的敲定还提前数年。(看到这样的结论，大家可能觉得自家团队的生产周期已经不算太慢了吧)

移动性

令工程师们不得不重新审视处理方式的要素还不仅仅在于基础硬件。多年来一直负责无线网络、移动网络以及移动相关问题的Jayanthi Rao就车载连接与无线技术的作用这一话题展开探讨。

当下车辆所使用的网络控制系统当中包含动力总成、底盘以及车门，全部都被整合到一套简单的广播网络当中。汽车系统的主要关注重点在于高可靠性——因此在99%的小数点后面再多加几个9自然是必须的，毕竟没人希望在驶下调整公路的时候行李箱被甩出车外，对吧？车辆控制信号则包含着数百万条代码。

除此之外，我们还希望在车中拥有更多连接类型：摄像头、GPS、蓝牙、Wi-Fi、GSM/LTE以及RFID。其中大部分都能在一定程度上实现外部设备连接——例如在我们携带移动设备进入车辆之后，仪表板会自动激活免提通话与音乐播放列表整合功能。每一种功能与车辆之间都存在内容共享协议——更多协议还将陆续出现。

交通运输行业中的开发工作一旦拥有标准化机制，其它各类创新成果也将应运而生，例如车辆与云服务之间进行对接。如何向车辆电池发出查询指

令以掌握其当前电量？当前位置距离商场还有多远？V2X标准——没错，另一项正在制定当中的标准——将解决车辆之间的通信难题，即V2V（车辆到车辆）服务与V2I（车辆到基础设施）服务。用再简明的说法来解释，这意味着车辆将能够与交通灯之间彼此交互，从而帮助我们的爱车更好地处理盲点警示、驾驶辅助请求、使用电子支付或者提供“周边地理位置”服务（例如通知朋友们‘跟着我的车一起来餐厅’）。

这方面面临的挑战之一在于车辆以高速行驶时，如何保证与其它系统间连接的可靠性。我们都知道，网络的设计初衷是应用在0英里每小时的环境下，Rao表示——连10英里每小时都未必能行，更不要说高速公路上的70英里每小时了。“这是一个根本性的问题，我们必须真正从根本上加以理解，”她表示。

哦，当然了——黑客活动也是客观存在的（有好有坏）。随着汽车领域开发工作的深入，黑客们也会找到自己新的可乘之机——其中包括允许开发人员编写更多软件从而最终与移动系统相整合的开源API。任何一套以性能表现与可靠性为出发点设计出的系统都会把注意力优先放在常规处理方面。在针对车辆（及代码）性能的开发工作中，安全举措（包括框架及指导方针）是为了实现性能目标而必须付出的成本。

娱乐性

不过我们大多数人都没有想到汽车连接会像如今这样被用户所使用。当然，作为消费者我们确实喜欢这些新奇的构思。“车辆正成为我们所持有过的最智能的消费级设备，”福特研究及创新中心数字化技术持有体验小组高级方案架构师Pramita Mitra指出，他的关注重点在于车载信息娱乐系统与云连接服务。Mitra援引GFK研究报告中的表述，称目前有五成左右的购买决策主要考虑车载信息娱乐（简称IVI）系统的效果，概括起来就是“让驾驶过程充满乐趣”。

这里说的可不是我年轻时流行的车载收音机（即使我对那个年代的老歌仍然很有感情）。IVI是一整套被整合到车辆当中的硬件设备集合，其中包括音响、导航、通讯以及车内环境控制系统。我们可以通过仪表板上的显示内容、多功能方向盘上的按键以及语音命令对其进行操作。想在车子的显示屏上查看最新影讯？体育赛事结果？想利用云服务远程控制我们的车载信息娱乐设定？这一切在如今的IVI系统硬件之上都已经可以实现。（这对于每一位技术人员都是个绝佳的发展机遇，我们需要考虑供应商与制造商如何为开发人员提供施展平台、又会拿出怎样的政策来吸引大家的加入。）

另外，我们还需要深入讨论用户界面的设计以及操作安全保障。（在时速70英里的情况下，大家还能安心完成多少操作？）

作为福特公司人机界面（简称HMI）事务的专家，Reates Curry与大部分同行人士一样正在借助虚拟测试系统（其中包括对司机可识别色域进行测试）了解用户与技术方案间的互动效果。值得一提的是，福特拥有一套名为VIRTTEX的大型驾驶模拟装置，其搭载360度全视野屏幕、转向反馈以及逼真的声音提示系统。这些将帮助用户体验专家了解那些听起来很酷的设计理念会给司机带来怎样的实际感受——不少点子虽然听起来很好，但在两年后就会由于用户的抱怨而被丢进垃圾桶。

大多数汽车厂商都是开发标准联盟的成员，会参与到汽车级功能相关法规与标准的制定当中。“我们必须在实际生产之前通过标准检验，”Curry

的态度还算积极。

职业生涯中的转折点

如果嵌入式系统、网络挑战、人机交互以及软件标准等领域带来的技术难题最终能以很酷的方式得到解决，那么必然要归功于出色的设计。

“我们技术小组的目标在于同与会者分享计算机科学家与IT专家在汽车行业中的成长经历，这种看似非传统的职业生涯规划往往会给很多具备相关知识背景的从业者带来巨大的发展机遇，”Mitra表示。

也许这将喜欢大家在“常规”软件世界之外，重新思索软件开发更为广泛的施展舞台。■

链接

微电影《Hello World—IT男的逆袭》

我决定还是由我自己来发这个微电影吧，自从拍了这个电影后，我在全国人民面前都抬不起头了，抓狂+崩溃。之前看到此电影的人大笑我转混娱乐圈，在这里我只想说一句：不准喷我，谁喷我，我就封谁账号 。

该片系由中国本土最大的开源技术社区——开源中国策划，由当红编剧@布小什大师 及@乔小囧担当编剧，看那些梦想着女神、财富、出人头地的IT屌丝男们如何逆袭！点击观看影片

大数据应用

追踪cookies已成往事？别高兴得太早

在谷歌、微软以及其它潜在厂商披露计划的具体细节之前，cookie仍将作为广告追踪的主要手段、我们对此也仍然有办法应付。如果最终cookie不可避免地走向历史舞台，我们将只能在众多新机制中做出选择。

微软与谷歌双双身陷传闻，据称正努力推出广告追踪技术的替代方案——但cookie的存在也许属于两害相权取其轻。

最近是不是要有点大动静？先是谷歌，现在又轮到微软，两大巨头身陷传闻，据称正努力推出新技术以取代以往在线广告服务所采取的cookie追踪机制。

关于二者项目的秘密倒没什么值得关注的，真正激起我们兴趣的内容在于，如果他们利用一套完全未知的方案取代目前这套众所周知的处理机制（当然也可以再加上‘天怒人怨’这个形容词），最终会引发怎样的结局？

关于谷歌“AdID”技术开始启动的说法源自今年九月，当时《今日美国》报道称“一种未知的广告识别机制……将取代第三方cookie”，从而实现对终端用户的广告追踪。据称，这套系统旨在“为浏览网页的消费者提供更多隐私空间与控制手段”，而且将被广泛用于“同意此项基本准则”的广告商——不过目前尚不清楚这些准则到底更倾向于保护消费者还是广告商。

这位来自谷歌公司内部的匿名线人并没有向《今日美国》透露更多细节信息，其中一部分原因在于这项建议将很快在“行业参与者、政府机构以及消费者组织之间进行传阅”。

谷歌公司在公布计划以及最终目标时，通常会引发一系列炒作活动。有些人认为谷歌可能希望采取类似于苹果通过其iAD平台在iOS上创建的方案达到目的——此方案2010年吸引到了来自美国监管机构的意外关注（苹果方案的曝光很可能促成了谷歌在同一时期对AdMob的收购交易）。

面对舆论对于更多细节的询问压力，一位“谷歌公司发言人”（根据多个网站的描述）仅仅回应称：“我们认为技术的改进能够在提升用户安全性的同时，确保网络体系仍然在经济层面上拥有可行性。我们与其它企业在这一领域拥有多种解决思路，但这些思路还处于早期筹备阶段。”

这里他提到了“其它企业”的说法，这很可能只是一种表达上的掩饰，但也不排除谷歌已经得到了相对准确的情报。

下面快进到今年十月，《广告时代》报道指出有传闻称微软也在打造自己的广告追踪技术（很可能同样采取无cookie路线）。业界普遍认为这一决定，无论实际情况如何，将与特定设备加以绑定——包括移动电话或者Xbox主机——从而对任

何经过标记或者与预设内容相关的行为数据直接由微软进行管理。（据称，这项技术在移动设备上的实际效果更为突出。）

前面提到的一切迹象都可能只是谷歌与微软丢给第三方厂商的一块骨头，而且引发了他们对于未来网络广告及营销业务的不安之感。努力创造一套不存在追踪行为的标准化方案并不能给我们带来与最初设想相符的隐私保护效果。在面对现实世界时，这类标准根本起不到什么实际作用。

有没有可能出现一种全新技术，既能为消费者提供必要的隐私保护、又能让广告商/营销人士获取生存所必需的情报？答案也许是肯定的；然而一旦新方案与老方案一样糟糕、甚至更糟，那么我们相当于绕了一圈又回到了原点。追踪cookie的做法虽然非常烦人，但至少这类行为很容易理解，而且围绕阻断这类追踪活动并对用户进行提醒已经催生出相当活跃的一整套次生行业。

PCWorld网站的Mark Hachman对于谷歌在追踪机制中抛弃cookie的做法感到不安，他提出了一些不同见解。“让我们假定自己能够忍受由cookie带来的各种不便，”他写道，即接受cookie中往往包含大量高度个人化的信息这一客观现实——更重要的是，这也是目前惟一一种能够被理解并认同的处理方式。一旦将其替换为其它更不透明的机制，结果必然令人担忧，尤其是在“谷歌转售用户个人数据、并以提供实用性服务的方式作为弥补”这一前提之下。

在谷歌、微软以及其它潜在厂商披露计划的具体细节之前，cookie仍将作为广告追踪的主要手段、我们对此也仍然有办法应付。如果最终cookie不可避免地走向历史舞台，我们将只能在众多新机制中做出选择。■

链 接

变懒的编程高手

年前，我还每天都阅读技术文章，有规律的观看演讲，大量的在stackoverflow上回答问题。后来我开始慢慢的减少这方面的 “努力”。只在微博上关注一些科技聚合信息源(HN，reddit，DZone)，这成了我唯一的浏览博客、文章的来源。我几乎完全停止了回答 stackoverflow上的问题。总之，我在抛弃那种经常的、超级活跃的参与科技世界活动的习惯，开始变成只在有需求时才去做的状态，只是偶尔阅读， 观看和回答问题。

为什么会这样？也许我开始认为自己是个“高手”了，大多数我遇到的东西要么是早已熟悉，要么是太小。一个新的web框架？ 一篇描述一种新的最佳实践方法的文章？我早已知道。一个非常特殊的用户案例？谷歌一下就能搜到我需要的文章。大多数在stackoverflow上的问题 都是老问题的重提。“如何解决这种错误”，“如何用这种框架完成这种任务”，“如何在Java里执行X”。

也许是被每小时里RSS阅读器里涌入的大量信息击倒，我最终停止使用RSS，转而使用微博，微博上的信息量只是我过去用RSS阅读的信息量的很少一部分。

这 是坏事吗？我会感觉跟不上最新科技趋势，慢慢失去对给定的问题采用正确的方法和技巧的能力吗？我不这么认为，至少目前不是。这是一个成熟的程序员的必经之 路吗？我不知道，但感觉符合逻辑——随着时间的积累，你开始对那些读到的文章挑三拣四。或是变懒了？或是过于自信？厌倦？或三者兼有？

我没有找到确定的答案，但至少目前感觉不错——我仍然在学习新技术，我仍然在关注科技最新动态。也许是我优化了学习方式。我现在只是要做到不那么入迷。我希望5年后不会变成一个中年的，懒惰的，过多自信的，愤世嫉俗的程序员，只要能像周围的人一样就好了。■

二维码的生成细节和原理

■ 这是一个私人谈话。我不单是对在这坐的Go开发团队成员说，我要感谢团队在推动Go发展上所做的一切。我还想感谢Go SF组织者给了我跟大家交流的机会。

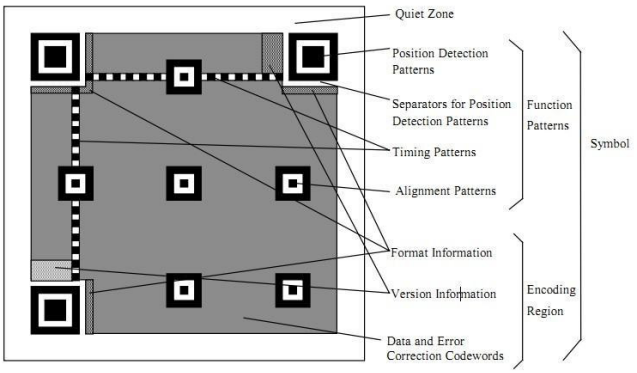
二维码又称QR Code，QR全称Quick Response，是一个近几年来移动设备上超流行的一种编码方式，它比传统的Bar Code条形码能存更多的信息，也能表示更多的数据类型：比如：字符，数字，日文，中文等等。这两天学习了一下二维码图片生成的相关细节，觉得这个玩意就 是一个密码算法，在此写一这篇文章 ，揭露一下。供好学的人一同学习之。

关于QR Code Specification，可参看这个PDF：http://raid-enii.net/files/datasheets/misc/qr_code.pdf

基础知识

首先，我们先说一下二维码一共有40个尺寸。官方叫版本Version。Version 1是21 x 21的矩阵，Version 2是 25 x 25的矩阵，Version 3是29的尺寸，每增加一个version，就会增加4的尺寸，公式是：(V-1)*4 + 21（V是版本号） 最高Version 40，(40-1)*4+21 = 177，所以最高是177 x 177 的正方形。

下面我们看看一个二维码的样例：



定位图案

◆ Position Detection Pattern是定位图案，用于标记二维码的矩形大小。这三个定位图案有白边叫Separators for Postion Detection Patterns。之所以三个而不是四个意思就是三个就可以标识一个矩形了。

◆ Timing Patterns也是用于定位的。原因是二维码有40种尺寸，尺寸过大了后需要有根标准线，不然扫描的时候可能会扫歪了。

◆ Alignment Patterns 只有Version 2以上（包括Version2）的二维码需要这个东东，同样是为了定位用的。

功能性数据

◆ Format Information 存在于所有的尺寸中，用于存放一些格式化数据的。

◆ Version Information 在 >= Version 7以上，需要预留两块3 x 6的区域存放一些版本信息。

数据码和纠错码

◆ 除了上述的那些地方，剩下的地方存放Data Code 数据码 和 Error Correction Code纠错码。

数据编码

我们先来说说数据编码。QR码支持如下编码：

Numeric mode 数字编码，从0到9。如果需要编码的数字的个数不是3的倍数，那么，最后剩下的1或2位数会被转成4或7bits，则其它的每3位数字会被编成 10，12，14bits，编成多长还要看二维码的尺寸（下面有一个表Table 3说明了这点）

Alphanumeric mode 字符编码。包括 0-9，大写的A到Z（没有小写），以及符号$ % * + - . / : 包括空格。这些字符会映射成一个字符索引表。如下所示：（其中的SP是空格，Char是字符，Value是其索引值） 编码的过程是把字符两两分组，然后转成下表的45进制，然后转成11bits的二进制，如果最后有一个落单的，那就转成6bits的二进制。而编码模式和 字符的个数需要根据不同的Version尺寸编成9，11或13个二进制（如下表中Table 3）

Char.	Value	Char.	Value	Char.	Value	Char.	Value	Char.	Value	Char.	Value	Char.	Value	Char.	Value
0	0	6	6	C	12	I	18	O	24	U	30	SP	36	.	42
1	1	7	7	D	13	J	19	P	25	V	31	$	37	/	43
2	2	8	8	E	14	K	20	Q	26	W	32	%	38	:	44
3	3	9	9	F	15	L	21	R	27	X	33	*	39		
4	4	A	10	G	16	M	22	S	28	Y	34	+	40		
5	5	B	11	H	17	N	23	T	29	Z	35	-	41		

Byte mode，字节编码，可以是0-255的ISO-8859-1字符。有些二维码的扫描器可以自动检测是否是UTF-8的编码。

Kanji mode 这是日文编码，也是双字节编码。同样，也可以用于中文编码。日文和汉字的编码会减去一个 值。如：在0X8140 to 0X9FFC中的字符会减去8140，在0XE040到0XEBBF中的字符要减去0XC140，然后把前两位拿出来乘以0XC0，然后再加上后两位，最 后转成13bit的编码。如下图示例：

Input character	“点”	“茗”
(Shift JIS value):	935F	E4AA
1. Subtract 8140 or C140	935F - 8140 = 121F	E4AA - C140 = 236A
2. Multiply m.s.b. by C0	12 × C0 = D80	23 × C0 = 1A40
3. Add l.s.b.	D80 + 1F = D9F	1A40 + 6A = 1AAA
4. Convert to 13 bit binary	0D9F → 0 1101 1001 1111	1AAA →1 1010 1010 1010

Extended Channel Interpretation (ECI) mode 主要用于特殊的字符集。并不是所有的扫描器都支持这种编码。

Structured Append mode 用于混合编码，也就是说，这个二维码中包含了多种编码格式。

FNC1 mode 这种编码方式主要是给一些特殊的工业或行业用的。比如GS1条形码之类的。

简单起见，后面三种不会在本文 中讨论。

下面两张表中，

◆ Table 2 是各个编码格式的“编号”，这个东西要写在Format Information中。注：中文是1101

◆ Table 3 表示了，不同版本（尺寸）的二维码，对于，数字，字符，字节和Kanji模式下，对于单个编码的2进制的位数。（在二维码的规格说明书中，有各种各样的编码规范表，后面还会提到）

Table 2 — Mode indicators

Mode	Indicator
ECI	**0111**
Numeric	**0001**
Alphanumeric	**0010**
8-bit Byte	**0100**
Kanji	**1000**
Structured Append	**0011**
FNC1	**0101** (First position) **1001** (Second position)
Terminator (End of Message)	**0000**

Table 3 — Number of bits in Character Count Indicator

Version	Numeric Mode	Alphanumeric Mode	8-bit Byte Mode	Kanji Mode
1 to 9	**10**	**9**	**8**	**8**
10 to 26	**12**	**11**	**16**	**10**
27 to 40	**14**	**13**	**16**	**12**

下面我们看几个示例，

示例一：数字编码

在Version 1的尺寸下，纠错级别为H的情况

下，编码： 01234567

把上述数字分成三组: 012 345 67

把他们转成二进制： 012 转成 0000001100 ； 345 转成 0101011001； 67 转成 1000011 。

把这三个二进制串起来：0000001100 0101011001 1000011

把数字的个数转成二进制 (version 1-H是10 bits): 8个数字的二进制是 0000001000

把数字编码的标志0001和第4步的编码加到前面： 0001 0000001000 0000001100 0101011001 1000011

示例二：字符编码

在Version 1的尺寸下，纠错级别为H的情况下，编码: AC-42

1. 从字符索引表中找到 AC-42 这五个字条的索引 (10,12,41,4,2)

2. 两两分组: (10,12) (41,4) (2)

3.把每一组转成11bits的二进制:

(10,12) 10*45+12 等于 462 转成 00111001110

(41,4) 41*45+4 等于 1849 转成 11100111001

(2) 等于 2 转成 000010

4. 把这些二进制连接起来：00111001110 11100111001 000010

5. 把字符的个数转成二进制 (Version 1-H为9 bits): 5个字符，5转成 000000101

6. 在头上加上编码标识 0010 和第5步的个数编码： 0010 000000101 00111001110 11100111001 000010

结束符和补齐符

假如我们有个HELLO WORLD的字符串要编码，根据上面的示例二，我们可以得到下面的编码，

编码	字符数	HELLO WORLD的编码
0010	000001011	01100001011 01111000110 10001011100 10110111000 10011010100 001101

我们还要加上结束符:

编码	字符数	HELLO WORLD的编码	结束
0010	000001011	01100001011 01111000110 10001011100 10110111000 10011010100 001101	0000

按8bits重排

如果所有的编码加起来不是8个倍数我们还要在后面加上足够的0，比如上面一共有78个bits，所以，我们还要加上2个0，然后按8个bits分好组:

00100000 01011011 00001011 01111000 11010001 01110010 11011100 01001101 01000011 01000000

补齐码（Padding Bytes）

最后，如果如果还没有达到我们最大的bits数的限制，我们还要加一些补齐码（Padding Bytes），Padding Bytes就是重复下面的两个

bytes：11101100 00010001 （这两个二进制转成十进制是236和17，我也不知道为什么，只知道Spec上是这么写的）关于每一个Version的每一种纠错级别的最大Bits限 制，可以参看QR Code Spec的第28页到32页的Table-7一表。

假设我们需要编码的是Version 1的Q纠错级，那么，其最大需要104个bits，而我们上面只有80个bits，所以，还需要24个bits，也就是需要3个Padding Bytes，我们就添加三个，于是得到下面的编码：

00100000 01011011 00001011 01111000 11010001 01110010 11011100 01001101 01000011 01000000 11101100 00010001 11101100

纠错码

上面我们说到了一些纠错级别，Error Correction Code Level，二维码中有四种级别的纠错，这就是为什么二维码有残缺还能扫出来，也就是为什么有人在二维码的中心位置加入图标。

错误修正容量	
L水平	7%的字码可被修正
M水平	15%的字码可被修正
Q水平	25%的字码可被修正
H水平	30%的字码可被修正

那么，QR是怎么对数据码加上纠错码的？首先，我们需要对数据码进行分组，也就是分成不同的Block，然后对各个Block进行纠错编码，对于如何分组，我们可以查看QR Code Spec的第33页到44页的Table-13到Table-22的定义表。注意最后两列：

Number of Error Code Correction Blocks：需要分多少个块。

Error Correction Code Per Blocks：每一个块中的code个数，所谓的code的个数，也就是有多少个8bits的字节。

5	134	L	26	1	(134,108,13)
		M	48	2	(67,43,12)
		Q	72	2 2	(33,15,9) (34,16,9)
		H	88	2 2	(33,11,11) (34,12,11)
6	172	L	36	2	(86,68,9)
		M	64	4	(43,27,8)
		Q	96	4	(43,19,12)
		H	112	4	(43,15,14)

[a] (c, k, r): c = total number of codewords
k = number of data codewords
r = number of error correction capacity

[b] Error correction capacity is less than half the number of error correction codewords to reduce the probability of misdecodes.

举个例子：上述的Version 5 + Q纠错级：需要4个Blocks（2个Blocks为一组，共两组），头一组的两个Blocks中各15个bits数据 + 各 9个bits的纠错码（注：表中的codewords就是一个8bits的byte）（再注：最后一例中的（c, k, r ）的公式为：c = k + 2 * r，因为后脚注解释了：纠错码的容量小于纠错码的一半）

下图给一个5-Q的示例（因为二进制写起来会让表格太大，所以，我都用了十进制）

组	块	数据	对每个块的纠错码
1	1	67 85 70 134 87 38 85 194 119 50 6 18 6 103 38	213 199 11 45 115 247 241 223 229 248 154 117 154 111 86 161 111 39
	2	246 246 66 7 118 134 242 7 38 86 22 198 199 146 6	87 204 96 60 202 182 124 157 200 134 27 129 209 17 163 163 120 133
2	1	182 230 247 119 50 7 118 134 87 38 82 6 134 151 50 7	148 116 177 212 76 133 75 242 238 76 195 230 189 10 108 240 192 141
	2	70 247 118 86 194 6 151 50 16 236 17 236 17 236 17 236	235 159 5 173 24 147 59 33 106 40 255 172 82 2 131 32 178 236

注：二维码的纠错码主要是通过Reed-Solomon error correction（里 德-所罗门纠错算法）来实现的。对于这个算法，对于我来说是

相当的复杂，里面有很多的数学计算，比如：多项式除法，把1-255的数映射成2的n次方（0<=n<=255）的伽罗瓦域Galois Field之类的神一样的东西，以及基于这些基础的纠错数学公式，因为我的数据基础差，对于我来说太过复杂，所以我一时半会儿还有点没搞明白，还在学习中，所以，我在这里就不展开说这些东西了。还请大家见谅了。（当然，如果有朋友很明白，也繁请教教我）

最终编码

穿插放置

如果你以为我们可以开始画图，你就错了。二维码的混乱技术还没有玩完，它还要把数据码和纠错码的各个codewords交替放在一起。如何交替呢，规则如下：

对于数据码：把每个块的第一个codewords先拿出来按顺度排列好，然后再取第一块的第二个，如此类推。如：上述示例中的Data Codewords如下：

块 1	67	85	70	134	87	38	85	194	119	50	6	18	6	103	38	
块 2	246	246	66	7	118	134	242	7	38	86	22	198	199	146	6	
块 3	182	230	247	119	50	7	118	134	87	38	82	6	134	151	50	7
块 4	70	247	118	86	194	6	151	50	16	236	17	236	17	236	17	236

我们先取第一列的：67，246，182，70

然后再取第二列的：67，246，182，70，85，246，230，247

如此类推：67，246，182，70，85，246，230，247 …… …… ，38，6，50，17，7，236

对于纠错码，也是一样：

块 1	213	199	11	45	115	247	241	223	229	248	154	117	154	111	86	161	111	39
块 2	87	204	96	60	202	182	124	157	200	134	27	129	209	17	163	163	120	133
块 3	148	116	177	212	76	133	75	242	238	76	195	230	189	10	108	240	192	141
块 4	235	159	5	173	24	147	59	33	106	40	255	172	82	2	131	32	178	236

和数据码取的一样，得到：213，87，148，235，199，204，116，159，…… …… 39，133，141，236

然后，再把这两组放在一起（纠错码放在数据码之后）得到：

67, 246, 182, 70, 85, 246, 230, 247, 70, 66, 247, 118, 134, 7, 119, 86, 87, 118, 50, 194, 38, 134, 7, 6, 85, 242, 118, 151, 194, 7, 134, 50, 119, 38, 87, 16, 50, 86, 38, 236, 6, 22, 82, 17, 18, 198, 6, 236, 6, 199, 134, 17, 103, 146, 151, 236, 38, 6, 50, 17, 7, 236, 213, 87, 148, 235, 199, 204, 116, 159, 11, 96, 177, 5, 45, 60, 212, 173, 115, 202, 76, 24, 247, 182, 133, 147, 241, 124, 75, 59, 223, 157, 242, 33, 229, 200, 238, 106, 248, 134, 76, 40, 154, 27, 195, 255, 117, 129, 230, 172, 154, 209, 189, 82, 111, 17, 10, 2, 86, 163, 108, 131, 161, 163, 240, 32, 111, 120, 192, 178, 39, 133, 141, 236

这就是我们的数据区。

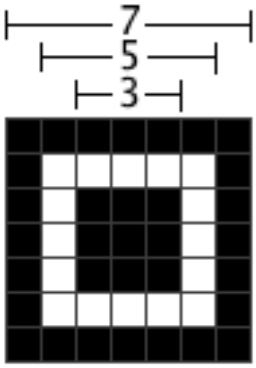
Remainder Bits

最后再加上Reminder Bits，对于某些Version的QR，上面的还不够长度，还要加上Remainder Bits，比如：上述的5Q版的二维码，还要加上7个bits，Remainder Bits加零就好了。关于哪些Version需要多少个Remainder bit，可以参看QR Code Spec的第15页的Table-1的定义表。

画二维码图

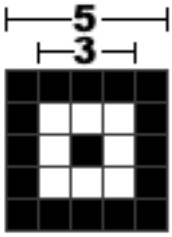
Position Detection Pattern

首先，先把Position Detection图案画在三个角上。（无论Version如何，这个图案的尺寸就是这么大）



Alignment Pattern

然后，再把Alignment图案画上（无论Version如何，这个图案的尺寸就是这么大）

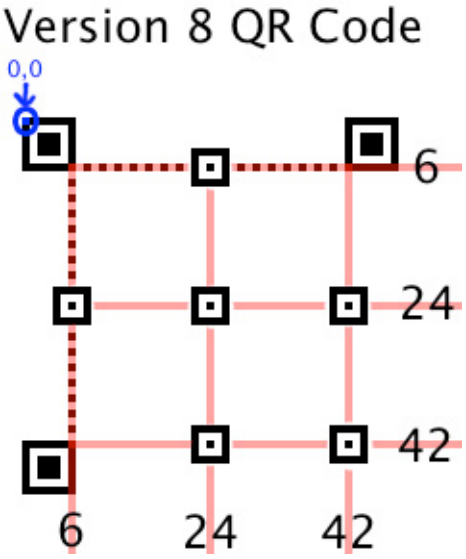


关于Alignment的位置，可以查看QR Code Spec的第81页的Table-E.1的定义表（下表是不完全表格）

Table E.1 — Row/column coordinates of center module of Alignment Patterns

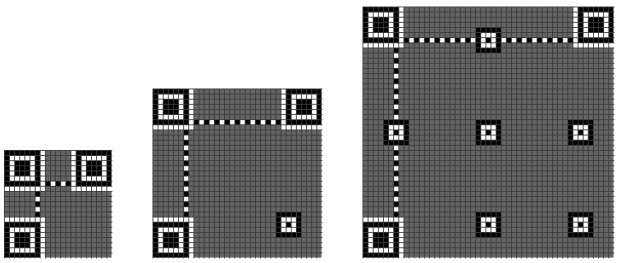
Version	Number of Alignment Patterns	Row/Column coordinates of center module						
1	0	-						
2	1	6	18					
3	1	6	22					
4	1	6	26					
5	1	6	30					
6	1	6	34					
7	6	6	22	38				
8	6	6	24	42				
9	6	6	26	46				
10	6	6	28	50				

下图是根据上述表格中的Version8的一个例子（6，24，42）



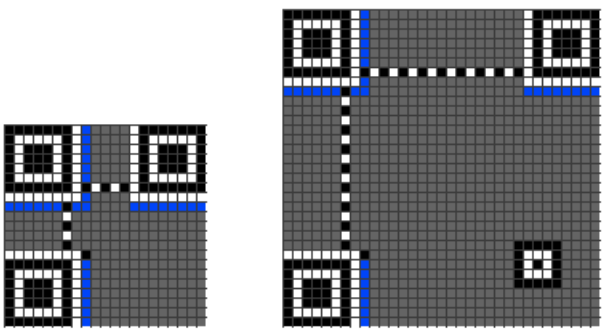
Timing Pattern

接下来是Timing Pattern的线（这个不用多说了）

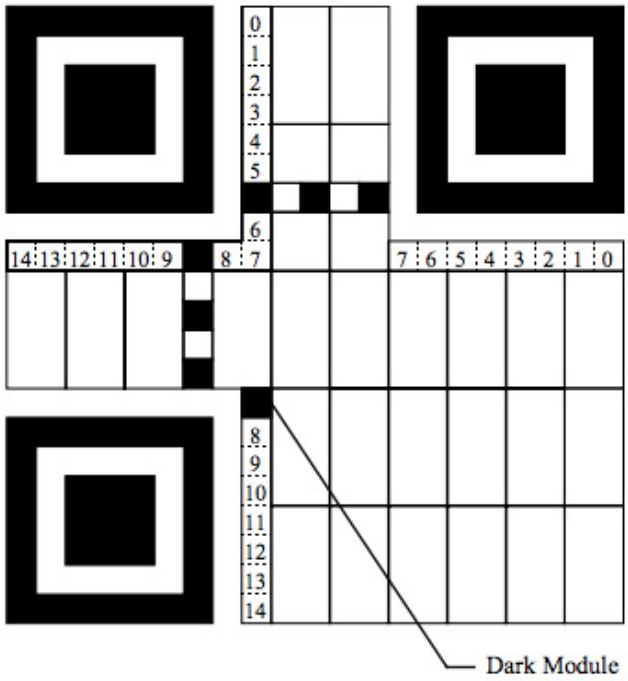


Format Information

再接下来是Formation Information，下图中的蓝色部分。



Format Information是一个15个bits的信息，每一个bit的位置如下图所示：（注意图中的Dark Module，那是永远出现的）



这15个bits中包括：

◆ 5个数据bits：其中，2个bits用于表示使用什么样的Error Correction Level， 3个bits表示使用什么样的Mask

◆ 10个纠错bits。主要通过BCH Code来计算

然后15个bits还要与101010000010010做XOR操作。这样就保证不会因为我们选用了00的纠错级别和000的Mask，从而造成全部为白色，这会增加我们的扫描器的图像识别的困难。

下面是一个示例：

Assume Error Correction Level M:	**00**
and Mask Pattern Reference:	**101**
Data:	**00101**
BCH bits:	**0011011100**
Unmasked bit sequence:	**001010011011100**
Mask pattern for XOR operation:	**101010000010010**
Format Information module pattern:	**100000011001110**

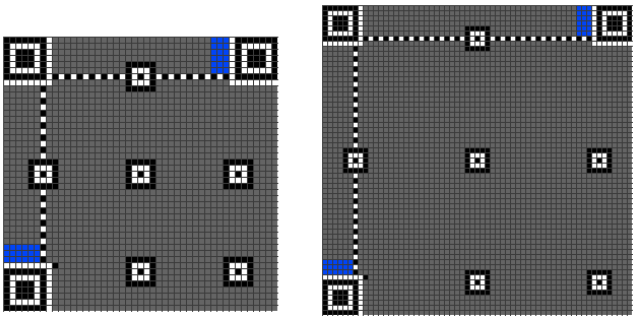
关于Error Correction Level如下表所示：

Error Correction Level	Binary indicator
L	01
M	00
Q	11
H	10

关于Mask图案如后面的Table 23所示。

Version Information

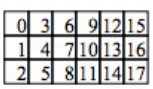
再接下来是Version Information（版本7以后需要这个编码），下图中的蓝色部分。



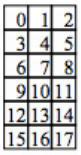
Version Information一共是18个bits，其中包括6个bits的版本号以及12个bits的纠错码，下面是一个示例：

Version number:	**7**
Data:	**000111**
BCH bits:	**110010010100**
Format Information module pattern:	**000111110010010100**

而其填充位置如下：



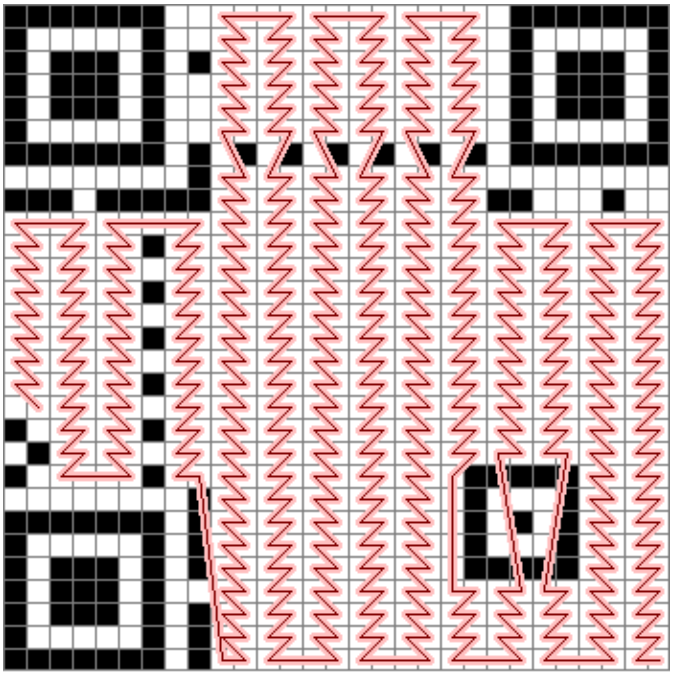
Version Information in lower left



Version Information in upper right

数据和数据纠错码

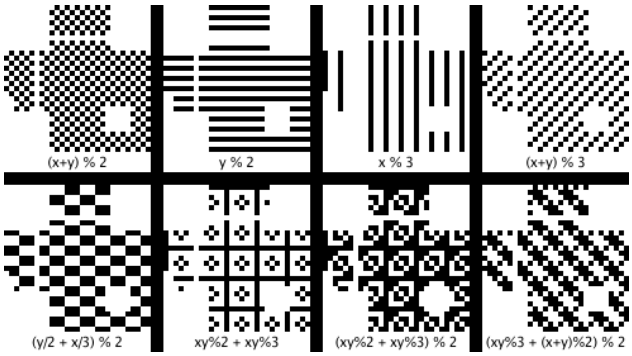
然后是填接我们的最终编码，最终编码的填充方式如下：从左下角开始沿着红线填我们的各个bits，1是黑色，0是白色。如果遇到了上面的非数据区，则绕开或跳过。



QR v3 order, from bottom right

掩码图案

这样下来，我们的图就填好了，但是，也许那些点并不均衡，如果出现大面积的空白或黑块，会告诉我们扫描识别的困难。所以，我们还要做Masking操作（靠，还嫌不复杂）QR的Spec中说了，QR有8个Mask你可以使用，如下所示：其中，各个mask的公式在各个图下面。所谓mask，说白了，就是和上面生成的图做XOR操作。Mask只会和数据区进行XOR，不会影响功能区。（注：选择一个合适的Mask也是有算法的）

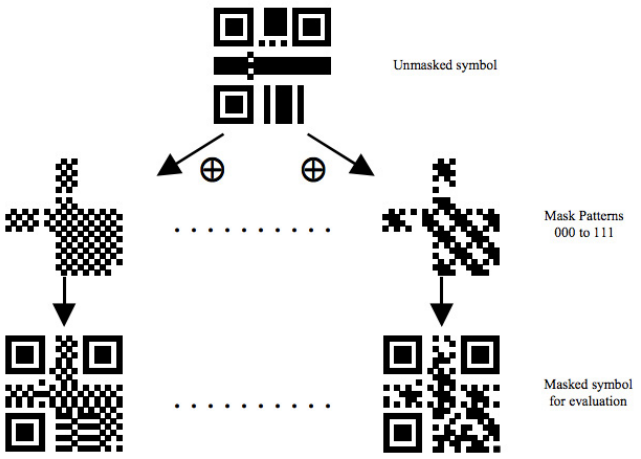


其Mask的标识码如下所示：（其中的i,j分别对应于上图的x,y）

Table 23 — Mask pattern generation conditions

Mask Pattern Reference	Condition
000	(*i* + *j*) mod 2 = 0
001	*i* mod 2 = 0
010	*j* mod 3 = 0
011	(*i* + *j*) mod 3 = 0
100	((*i* div 2) + (*j* div 3)) mod 2 = 0
101	(*i j*) mod 2 + (*i j*) mod 3 = 0
110	((*i j*) mod 2 + (*i j*) mod 3) mod 2 = 0
111	((*i j*) mod 3 + (*i*+*j*) mod 2) mod 2 = 0

下面是Mask后的一些样子，我们可以看到被某些Mask XOR了的数据变得比较零散了。



Mask过后的二维码就成最终的图了。

好了，大家可以去尝试去写一下QR的编码程序，当然，你可以用网上找个Reed Soloman的纠错算法的库，或是看看别人的源代码是怎么实现这个繁锁的编码。■

编者按

当我们感觉自己很擅长一件事的时候，才会真正地去学习它，花费大量的时间和精力，全身心投入，直到非常精通为止。

由12306.cn谈谈网站性能技术

12306.cn网站挂了，被全国人民骂了。我这两天也在思考这个事，我想以这个事来粗略地和大家讨论一下网站性能的问题。因为仓促，而且完全基于本人有限的经验和了解，所以，如果有什么问题还请大家一起讨论和指正。（这又是一篇长文，只讨论性能问题，不讨论那些UI，用户体验，或是是否把支付和购票下单环节分开的功能性的东西）

业务

任何技术都离不开业务需求，所以，要说明性能问题，首先还是想先说说业务问题。

其一，有人可能把这个东西和QQ或是网游相比。但我觉得这两者是不一样的，网游和QQ在线或是登录时访问的更多的是用户自己的数据，而订票系统访问的是中心的票量数据，这是不一样的。不要觉得网游或是QQ能行你就以为这是一样的。网游和QQ 的后端负载相对于电子商务的系统还是简单。

其二，有人说春节期间订火车的这个事好像网站的秒杀活动。的确很相似，但是如果你的思考不在表面的话，你会发现这也有些不一样。火车票这个事，还有很多查询操作，查时间，查座位，查铺位，一个车次不 行，又查另一个车次，其伴随着大量的查询操作，下单的时候需要对数据库操作。而秒杀，直接杀就好了。另外，关于秒杀，完全可以做成只接受前N个用户的请求（完全不操作后端的任何数据， 仅仅只是对用户的下单操作log），这种业务，只需要在内存cache中放好可秒杀的数量，还可以把数据分布开来放，100商品，10台服务器一台放10个，无需在当时操作任何数据库。可以订单数够后，停止秒杀，然后批量写数据库。而且秒杀的商品不多。火车票这个不是像秒杀那么简单的，春运时间，几乎所有的票都是热门票，而且几乎是全国人民都来了。（淘宝的双十一也就3百万用户，而火车票瞬时有千万级别甚至是亿级别的）

其三，有人拿这个系统和奥运会的票务系统比较。我觉得还是不一样。虽然奥运会的票务系统当年也一上线就废了。但是奥运会用的是抽奖的方式，也就是说不存在先来先得的抢的方式，而且，是事后抽奖，事前只需要收信息，事前不需要保证数据一致性，没有锁，很容易水平扩展。

其四，订票系统应该和电子商务的订单系统很相似，都是需要对库存进行：1）占住库存，2）支付（可选），3）扣除库存的操作。这个是需要有一致性的检查的，也就是在并发时需要对数据加锁的。B2C的电商基本上都会把这个事干成异步的，也就是说，你下的订单并不是马上处理的，而是延时处理的，只有成功处理了，系统才会给你一封确

认邮件说是订单成功。我相信有很多朋友都收到认单不成功的邮件。这就是说，数据一致性在并发下是一个瓶颈。

其五，铁路的票务业务很变态，其采用的是突然放票，而有的票又远远不够大家分，所以，大家才会有抢票这种有中国特色的业务的做法。于是当票放出来的时候，就会有几百万人甚至上千万人杀上去，查询，下单。几十分钟内，一个网站能接受几千万的访问量，这个是很恐怖的事情。据说12306的高峰访问是10亿PV，集中在早8点到10点，每秒PV在高峰时上千万。

多说几句：

库存是B2C的恶梦，库存管理相当的复杂。不信，你可以问问所有传统和电务零售业的企业，看看他们管理库存是多么难的一件事。不然，就不会有那么多人在问凡客的库存问题了。（你还可以看看《乔布斯传》，你就知道为什么Tim会接任Apple的CEO了，最主要的原因是他搞定了苹果的库存周期问题）

对于一个网站来说，浏览网页的高负载很容易搞定，查询的负载有一定的难度去处理，不过还是可以通过缓存查询结果来搞定，最难的就是下单的负载。因为要访问库存啊，对于下单，基本上是用异步来搞定的。去年双11节，淘宝的每小时的订单数大约在60万左右，京东一天也才能支持40万（居然比12306还差），亚马逊5年前一小时可支持70万订单量。可见，下订单的操作并没有我们相像的那么性能高。

淘宝要比B2C的网站要简单得多，因为没有仓库，所以，不存在像B2C这样有N个仓库对同一商品库存更新和查询的操作。下单的时候，B2C的 网站要去找一个仓库，又要离用户近，又要有库存，这需要很多计算。试想，你在北京买了一本书，北京的仓库没货了，就要从周边的仓库调，那就要去看看沈阳或 是西安的仓库有没有货，如果没有，又得看看江苏的仓库，等等。淘宝的就没有那么多事了，每个商户有自己的库存，库存就是一个数字，并且库存分到商户头上了，反而有利于性能扩展。

数据一致性才是真正的性能瓶颈。有 人说nginx可以搞定每秒10万的静态请求，我不怀疑。但这只是静态请求，理论值，只要带宽、I/O够强，服务器计算能力够，并支持的并发连接数顶得住10万TCP链接的建立 的话，那没有问题。但在数据一致性面前，这10万就完完全全成了一个可望不可及的理论值了。

我说那么多，我只是想从业务上告诉大家，我们需要从业务上真正了解春运铁路订票这样业务的变态之处。

前端性能优化技术

要解决性能的问题，有很多种常用的方法，我在下面列举一下，我相信12306这个网站使用下面的这些技术会让其性能有质的飞跃。

一、前端负载均衡

通过DNS的负载均衡器（一般在路由器上根据路由的负载重定向）可以把用户的访问均匀地分散在多个Web服务器上。这样可以减少Web服务器的请求负载。因为http的请求都是短作业，所以，可以通过很简单的负载均衡器来完成这一功能。最好是有CDN网络让用户连接与其最近的服务器（CDN通常伴随着分布式存储）。（关于负载均衡更为详细的说明见“后端的负载均衡”）

二、减少前端链接数

我看了一下12306.cn，打开主页需要建60多个HTTP连接，车票预订页面则有70多个HTTP请求，现在的浏览器都是并发请求的（当然，浏览器的一个页面的并发数是有限的，但是你挡不住用户开多个页面，而且，后端服务器TCP链接在前端断开始，还不会马上释放或重要）。所以，只要有100万个用户，就有可能会有6000万个链接（访问第一次后有了浏览器端的cache，这个数会下来，就算只有20%也是百万级的链接数），太多了。一个登录查询页面就好了。把js打成一个文件，把css也打成一个文件，把图标也打成一个文件，用css分块展示。把链接数减到最低。

三、减少网页大小增加带宽

这个世界不是哪个公司都敢做图片服务的，因为图片太耗带宽了。现在宽带时代很难有人能体会到当拨号时代做个图页都不敢用图片的情形（现在在手机端浏览也是这个情形）。我查看了一下12306首页的需要下载的总文件大小大约在900KB左右，如果你访问过了，浏览器会帮你缓存很多，只需下载10K左右的文件。但是我们可以想像一个极端一点的案例，1百万用户同时访问，且都是第一次访问，每人下载量需要1M，如果需要在120秒内返回，那么就需要，1M * 1M /120 * 8 = 66Gbps的带宽。很惊人吧。所以，我估计在当天，12306的阻塞基本上应该是网络带宽，所以，你可能看到的是没有响应。后面随着浏览器的缓存帮助12306减少很多带宽占用，于是负载一下就到了后端，后端的数据处理瓶颈一下就出来。于是你会看到很多http 500之类的错误。这说明后端服务器垮了。

四、前端页面静态化

静态化一些不常变的页面和数据，并gzip一下。还有一个变态的方法是把这些静态页面放在/dev/shm下，这个目录就是内存，直接从内存中把文件读出来返回，这样可以减少昂贵的磁盘I/O。使用nginx的sendfile功能可以让这些静态文件直接在内核心态交换，可以极大增加性能。

五、优化查询

很多人查询都是在查一样的，完全可以用反向代理合并这些并发的相同的查询。这样的技术主要用查询结果缓存来实现，第一次查询走数据库获得数据，并把数据放到缓存，后面的查询统统直接访问高速缓存。为每个查询做Hash，使用NoSQL的技术可以完成这个优化。（这个技术也可以用做静态页面）

对于火车票量的查询，个人觉得不要显示数字，就显示一个“有”或“无”就好了，这样可以大大简化系统复杂度，并提升性能。把查询对数据库的负载分出去，从而让数据库可以更好地为下单的人服务。

六、缓存的问题

缓存可以用来缓存动态页面，也可以用来缓存查询的数据。缓存通常有那么几个问题：

1）缓存的更新。也叫缓存和数据库的同步。有这么几种方法，一是缓存time out，让缓存失效，重查，二是，由后端通知更新，一量后端发生变化，通知前端更新。前者实现起来比较简单，但实时性不高，后者实现起来比较复杂，但实时性高。

2）缓存的换页。内存可能不够，所以，需要把

一些不活跃的数据换出内存，这个和操作系统的内存换页和交换内存很相似。FIFO、LRU、LFU都是比较经典的换页算法。相关内容参看Wikipeida的缓存算法。

3）缓存的重建和持久化。缓存在内存，系统总要维护，所以，缓存就会丢失，如果缓存没了，就需要重建，如果数据量很大，缓存重建的过程会很慢，这会影响生产环境，所以，缓存的持久化也是需要考虑的。

诸多强大的NoSQL都很好支持了上述三大缓存的问题。

后端性能优化技术

前面讨论了前端性能的优化技术，于是前端可能就不是瓶颈问题了。那么性能问题就会到后端数据上来了。下面说几个后端常见的性能优化技术。

一、数据冗余

关于数据冗余，也就是说，把我们的数据库的数据冗余处理，也就是减少表连接这样的开销比较大的操作，但这样会牺牲数据的一致性。风险比较大。很多人把NoSQL用做数据，快是快了，因为数据冗余了，但这对数据一致性有大的风险。这需要根据不同的业务进行分析和处理。（注意：用关系型数据库很容易移植到NoSQL上，但是反过来从NoSQL到关系型就难了）

二、数据镜像

几乎所有主流的数据库都支持镜像，也就是replication。数据库的镜像带来的好处就是可以做负载均衡。把一台数据库的负载均分到多台上，同时又保证了数据一致性（Oracle的SCN）。最重要的是，这样还可以有高可用性，一台废了，还有另一台在服务。

数据镜像的数据一致性可能是个复杂的问题，所以我们要在单条数据上进行数据分区，也就是说，把一个畅销商品的库存均分到不同的服务器上，如，一个畅销商品有1万的库存，我们可以设置10台服务器，每台服务器上有1000个库存，这就好像B2C的仓库一样。

三、数据分区

数据镜像不能解决的一个问题就是数据表里的记录太多，导致数据库操作太慢。所以，把数据分区。数据分区有很多种做法，一般来说有下面这几种：

1）把数据把某种逻辑来分类。比如火车票的订票系统可以按各铁路局来分，可按各种车型分，可以按始发站分，可以按目的地分……，反正就是把一张表拆成多张有一样的字段但是不同种类的表，这样，这些表就可以存在不同的机器上以达到分担负载的目的。

2）把数据按字段分，也就是竖着分表。比如把一些不经常改的数据放在一个表里，经常改的数据放在另外多个表里。把一张表变成1对1的关系，这样，你可以减少表的字段个数，同样可以提升一定的性能。另外，字段多会造成一条记录的存储会被放到不同的页表里，这对于读写性能都有问题。但这样一来会有很多复杂的控制。

3）平均分表。因为第一种方法是并不一定平均分均，可能某个种类的数据还是很多。所以，也有采用平均分配的方式，通过主键ID的范围来分表。

4）同一数据分区。这个在上面数据镜像提过。也就是把同一商品的库存值分到不同的服务器上，

比如有10000个库存，可以分到10台服务器上，一台上有1000个库存。然后负载均衡。

这三种分区都有好有坏。最常用的还是第一种。数据一旦分区，你就需要有一个或是多个调度来让你的前端程序知道去哪里找数据。把火车票的数据分区，并放在各个省市，会对12306这个系统有非常有意义的质的性能的提高。

四、后端系统负载均衡

前面说了数据分区，数据分区可以在一定程度上减轻负载，但是无法减轻热销商品的负载，对于火车票来说，可以认为是大城市的某些主干线上的车票。这就需要使用数据镜像来减轻负载。使用数据镜像，你必然要使用负载均衡，在后端，我们可能很难使用像路由器上的负载均衡器，因为那是均衡流量的，因为流量并不代表服务器的繁忙程度。因此，我们需要一个任务分配系统，其还能监控各个服务器的负载情况。

任务分配服务器有一些难点：

负载情况比较复杂。什么叫忙？是CPU高？还是磁盘I/O高？还是内存使用高？还是并发高？还是内存换页率高？你可能需要全部都要考虑。这些信息要发送给那个任务分配器上，由任务分配器挑选一台负载最轻的服务器来处理。

任务分配服务器上需要对任务队列，不能丢任务啊，所以还需要持久化。并且可以以批量的方式把任务分配给计算服务器。

任务分配服务器死了怎么办？这里需要一些如Live-Standby或是failover等高可用性的技术。我们还需要注意那些持久化了的任务的队列如何转移到别的服务器上的问题。

我看到有很多系统都用静态的方式来分配，有的用hash，有的就简单地轮流分析。这些都不够好，一个是不能完美地负载均衡，另一个静态的方法的致命缺陷是，如果有一台计算服务器死机了，或是我们需要加入新的服务器，对于我们的分配器来说，都需要知道的。另外，还要重算哈希（一致性hash可以部分解决这个问题）。

还有一种方法是使用抢占式的方式进行负载均衡，由下游的计算服务器去任务服务器上拿任务。让这些计算服务器自己决定自己是否要任务。这样的好处是可以简化系统的复杂度，而且还可以任意实时地减少或增加计算服务器。但是唯一不好的就是，如果有一些任务只能在某种服务器上处理，这可能会引入一些复杂度。不过总体来说，这种方法可能是比较好的负载均衡。

五、异步、 throttle 和 批量处理

异步、throttle（节流阀） 和批量处理都需要对并发请求数做队列处理的。

异步在业务上一般来说就是收集请求，然后延时处理。在技术上就是可以把各个处理程序做成并行的，也就可以水平扩展了。但是异步的技术问题大概有这些，a）被调用方的结果返回，会涉及进程线程间通信的问题。b）如果程序需要回滚，回滚会有点复杂。c）异步通常都会伴随多线程多进程，并发的控制也相对麻烦一些。d）很多异步系统都用消息机制，消息的丢失和乱序也会是比较复杂的问题。

throttle 技术其实并不提升性能，这个技术主要是防止系统被超过自己不能处理的流量给搞垮了，这其实是个保护机制。使用throttle技术一般来说是对于一些自己无法控制的系统，比如，和你

网站对接的银行系统。

批量处理的技术，是把一堆基本相同的请求批量处理。比如，大家同时购买同一个商品，没有必要你买一个我就写一次数据库，完全可以收集到一定数量的请求，一次操作。这个技术可以用作很多方面。比如节省网络带宽，我们都知道网络上的MTU（最大传输单元），以态网是1500字节，光纤可以达到4000多个字节，如果你的一个网络包没有放满这个MTU，那就是在浪费网络带宽，因为网卡的驱动程序只有一块一块地读效率才会高。因此，网络发包时，我们需要收集到足够多的信息后再做网络I/O，这也是一种批量处理的方式。批量处理的敌人是流量低，所以，批量处理的系统一般都会设置上两个阀值，一个是作业量，另一个是timeout，只要有一个条件满足，就会开始提交处理。

所以，只要是异步，一般都会有throttle机制，一般都会有队列来排队，有队列，就会有持久化，而系统一般都会使用批量的方式来处理。

云风同学设计的“排队系统” 就是这个技术。这和电子商务的订单系统很相似，就是说，我的系统收到了你的购票下单请求，但是我还没有真正处理，我的系统会跟据我自己的处理能力来throttle住这些大量的请求，并一点一点地处理。一旦处理完成，我就可以发邮件或短信告诉用户你来可以真正购票了。

在这里，我想通过业务和用户需求方面讨论一下云风同学的这个排队系统，因为其从技术上看似解决了这个问题，但是从业务和用户需求上来说可能还是有一些值得我们去深入思考的地方：

1）队列的DoS攻击。首先，我们思考一下，这个队是个单纯地排队的吗？这样做还不够好，因为这样我们不能杜绝黄牛，而且单纯的ticket_id很容易发生DoS攻击，比如，我发起N个 ticket_id，进入购票流程后，我不买，我就耗你半个小时，很容易我就可以让想买票的人几天都买不到票。有人说，用户应该要用身份证来排队， 这样在购买里就必需要用这个身份证来买，但这也还不能杜绝黄牛排队或是号贩子。因为他们可以注册N个帐号来排队，但就是不买。黄牛这些人这个时候只需要干一个事，把网站搞得正常人不能访问，让用户只能通过他们来买。

2）对列的一致性？对这个队列的操作是不是需要锁？只要有锁，性能一定上不去。试想，100万个人同时要求你来分配位置号，这个队列将会成为性能瓶颈。你一定没有数据库实现得性能好，所以，可能比现在还差。抢数据库和抢队列本质上是一样的。

3）队列的等待时间。购票时间半小时够不够？多不多？要是那时用户正好不能上网呢？如果时间短了，用户不够时间操作也会抱怨，如果时间长了，后面在排队的那些人也会抱怨。这个方法可能在实际操作上会有很多问题。另外，半个小时太长了，这完全不现实，我们用15分钟来举例：有1千万用户，每一个时刻只能放进去1万个，这1万个用户需要15分钟完成所有操作，那么，这1千万用户全部处理完，需要1000*15m = 250小时，10天半，火车早开了。（我并非信口开河，根据铁道部专家的说明：这几天，平均一天下单100万，所以，处理1000万的用户需要十天。这个计算可能有点简单了，我只是想说，在这样低负载的系统下用排队可能都不能解决业务问题）

4）队列的分布式。这个排队系统只有一个队列好吗？还不足够好。因为，如果你放进去的可以购票的人如果在买同一个车次的同样的类型的票（比如某动车卧铺），还是等于在抢票，也就是说系统的负载还是会有可能集中到其中某台服务器上。因此，最好的方法是根据用户的需求——提供出发地和目的地，来对用户进行排队。而这样一来，队列也就可以是多个，只要是多个队列，就可以水平扩展了。这样可以解决性能问题，但是没有解决用户长时间排队的问题。

我觉得完全可以向网上购物学习。在排队（下单）的时候，收集好用户的信息和想要买的票，并允许用户设置购票的优先级，比如，A车次卧铺买 不到就买 B车次的卧铺，如果还买不到就买硬座等等，然后用户把所需的钱先充值好，接下来就是系统完全自动地异步处理订单。成功不成功都发短信或邮件通知用户。这样，系统不仅可以省去那半个小时的用户交互时间，自动化加快处理，还可以合并相同购票请求的人，进行批处理（减少数据库的操作次数）。这种方法最妙的事是可以知道这些排队用户的需求，不但可以优化用户的队列，把用户分布到不同的队列，还可以像亚马逊的心愿单一样，通过一些计算就可以让铁道部做车次统筹安排和调整（最后，排队系统（下单系统）还是要保存在数据库里的或做持久化，不能只放在内存中，不然机器一down，就等着被骂吧）。

小结

写了那么多，我小结一下：

0）无论你怎么设计，你的系统一定要能容易地水平扩展。也就是说，你的整个数据流中，所有的环节都要能够水平扩展。这样，当你的系统有性能问题时，“加30倍的服务器”才不会被人讥笑。

1）上述的技术不是一朝一夕能搞定的，没有长期的积累，基本无望。我们可以看到，无论你用哪种都会引发一些复杂性，设计总是在做一种权衡。

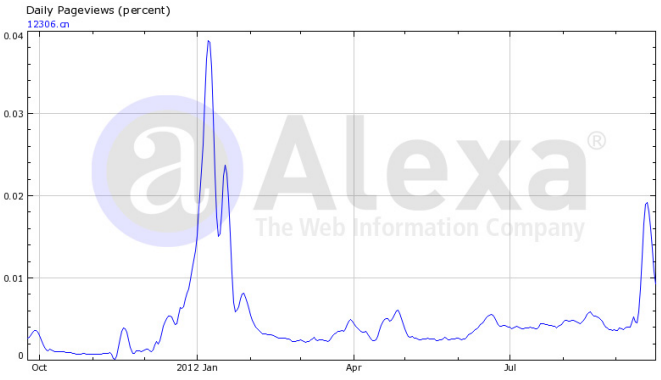
2）集中式的卖票很难搞定，使用上述的技术可以让订票系统能有几佰倍的性能提升。而在各个省市建分站，分开卖票，是能让现有系统性能有质的提升的最好方法。

3）春运前夕抢票且票量供远小于求这种业务模式是相当变态的，让几千万甚至上亿的人在某个早晨的8点钟同时登录同时抢票的这种业务模式是变态中的变态。业务形态的变态决定了无论他们怎么办干一定会被骂。

4）为了那么一两个星期而搞那么大的系统，而其它时间都在闲着，有些可惜了，这也就是铁路才干得出来这样的事了。

更新2012年9月27日

Alexa 统计的12306的PV （注：Alexa的PV定义是：一个用户在一天内对一个页面的多次点击只算一次）■



十种更好的表达“你的代码写的很烂”的方法

□ 外刊IT评论 /文

如果你有一个同事，他写的程序与其说是代码，不如说更像希腊神话中女妖美杜莎的头发，你当然不能熟视无睹，你应该做出一些反应，但你可选的合适的反 应方式并没有多少：自己默默的帮他整理清楚、向上级抱怨、向其他同事背后唠叨此事、闷在心里直到憋不住，或者这最大胆的方法：走上去直接对烂程序员说他的 代码很烂。

事实上，这最大胆的方法其实也是最好的方法。大多时候，你可以做的巧妙些，从而避免由此引起的感情伤害或引发咆哮比赛。就像一句古话：只要方式正确，你可以向一个人说任何话。

当然，找到这正确的方式并不是轻而易举的事情。为了方法大家行事，下面是10种让你的表达更具技巧性的好方法。

开门见山：告诉他你看不懂他写的代码，并追加一些像这样的话：“我需要你帮我理解这块代码”——这是“硅谷iOS程序员研讨会”组织者、软件程序员Tim Burks的话。

推心置腹：约他出去喝两瓶啤酒，麻痹他的抵抗情绪，先从讨论编码风格说起。你会发现，他之所以这样写代码是因为这样他很方便——而不是方便开发团队。通过讨论代码不仅仅是人和机器交流的工具，更重要的是通过代码的人和人的交流，你可以让他用一种全新的思维来认识代码。

高山仰止：如果你的同事敬重你，想必他也会敬仰或效仿你所敬仰的著名程序员。所以，跟他讲那些杰出程序员的故事。或者向他转述Burks的观察所得：杰出的程序员总能把自己的编码风格融入到他人的风格中。

一针见血：Adobe System研究实验室的领袖人物Tom Jacobs说，“为了格式而格式化代码毫无意义，但将调整代码格式作为重构工作的一部分，增加新功能、修改bug工作的一部分，那是很正常的，因为这样做本质的增加了代码的质量。”

反馈问题，而不是批评：心理学家Leon Seltzer在“当代心理学”上的一篇博客中说，“人们更喜欢接受反馈信息而不是批评——即使是负面反馈”。所以，以反馈问题的形式诉说问题。

以后改进：不要苛求当前的工作，而是要求日后对此改进提高。按这种思路，你可以说：“嗨，下一次，如果你能把每个方法的行数减到10行以下，那会更好。”这比说“你的代码一塌糊涂”要中听的多。

糖衣炮弹：封装你的批评，在表达“你的代码很烂”的意思前和后先恭维一番。

偷换概念：如果交谈中总是说你、你、你，这很容易引起敌意，就好象你在指控罪名。所以，不如换种方式，与其说“每次都让我为你写的代码擦屁股”，不如说“有时候我真感到很沮丧，因为需要重你的代码。”

引蛇出洞：这种办法稍微有些麻烦，但不失为一种以守为攻的好办法。组织一些编程大赛之类的活动。如果顺利的话，它能引出一场安全的、没有猜疑的关于如何提高你的同事的代码质量的讨论。■

链接

Winamp 曾经我们最爱的播放器将于下月关闭

Winamp 现在归属于 AOL，这个昔时非常受欢迎的 MP3 播放器，将于下月 20 日宣布关闭其商店服务（winamp.com）。也就是说你还有一个月时间来使用它的服务。当然你还可以继续使用 Winamp 这个软件。这个通知让我们很诧异，但或许也只是我们有所怀旧的因素。毕竟我们都已经快要忘却这个软件了。

要不要下载一个留作纪念呢?

http://www.winamp.com/media-player/en

专题推荐

Nginx 战斗准备 —— 优化指南

□ 开源中国 / 编译

大多数的Nginx安装指南告诉你如下基础知识——通过apt-get安装，修改这里或那里的几行配置，好了，你已经有了一个Web服务器了！而且，在大多数情况下，一个常规安装的nginx对你的网站来说已经能很好地工作了。然而，如果你真的想挤压出nginx的性能，你必须更深入一些。在本指南中，我将解释Nginx的那些设置可以微调，以优化处理大量客户端时的性能。需要注意一点，这不是一个全面的微调指南。这是一个简单的预览——那些可以通过微调来提高性能设置的概述。你的情况可能不同。

基本的 (优化过的)配置

我们将修改的唯一文件是nginx.conf，其中包含Nginx不同模块的所有设置。你应该能够在服务器的/etc/nginx目录中找到nginx.conf。首先，我们将谈论一些全局设置，然后按文件中的模块挨个来，谈一下哪些设置能够让你在大量客户端访问时拥有良好的性能，为什么它们会提高性能。本文的结尾有一个完整的配置文件。

高层的配置

nginx.conf文件中，Nginx中有少数的几个高级配置在模块部分之上。

```
1. user www-data;
2. pid /var/run/nginx.pid;
3. worker_processes auto;
4. worker_rlimit_nofile 100000;
```

user和pid应该按默认设置 - 我们不会更改这些内容，因为更改与否没有什么不同。

worker_processes 定义了nginx对外提供web服务时的worder进程数。最优值取决于许多因素，包括（但不限于）CPU核的数量、存储数据的硬盘数量及负载模式。不能确定的时候，将其设置为可用的CPU内核数将是一个好的开始（设置为“auto”将尝试自动检测它）。

worker_rlimit_nofile 更改worker进程的最大打开文件数限制。如果没设置的话，这个值为操作系统的限制。设置后你的操作系统和Nginx可以处理比“ulimit -a”更多的文件，所以把这个值设高，这样nginx就不会有“too many open files”问题了。

Events模块

events模块中包含nginx中所有处理连接的设置。

```
1. events {
2.     worker_connections 2048;
3.     multi_accept on;
4.     use epoll;
5. }
```

worker_connections设置可由一个worker进程同时打开的最大连接数。如果设置了上面提到的

worker_rlimit_nofile，我们可以将这个值设得很高。

记住，最大客户数也由系统的可用socket连接数限制（~ 64K），所以设置不切实际的高没什么好处。

multi_accept 告诉nginx收到一个新连接通知后接受尽可能多的连接。

use 设置用于复用客户端线程的轮询方法。如果你使用Linux 2.6+，你应该使用epoll。如果你使用*BSD，你应该使用kqueue。想知道更多有关事件轮询？看下维基百科吧（注意，想了解一切的话可能需要neckbeard和操作系统的课程基础）

（值得注意的是如果你不知道Nginx该使用哪种轮询方法的话，它会选择一个最适合你操作系统的）

HTTP 模块

HTTP模块控制着nginx http处理的所有核心特性。因为这里只有很少的配置，所以我们只节选配置的一小部分。所有这些设置都应该在http模块中，甚至你不会特别的注意到这段设置。

```
1. http {
2.     server_tokens off;
3.     sendfile on;
4.
5.     tcp_nopush on;
6.     tcp_nodelay on;
7.     ...
8. }
```

server_tokens 并不会让nginx执行的速度更快，但它可以关闭在错误页面中的nginx版本数字，这样对于安全性是有好处的。

sendfile可以让sendfile()发挥作用。sendfile()可以在磁盘和TCP socket之间互相拷贝数据(或任意两个文件描述符)。Pre-sendfile是传送数据之前在用户空间申请数据缓冲区。之后用read()将数据从文件拷贝到这个缓冲区，write()将缓冲区数据写入网络。sendfile()是立即将数据从磁盘读到OS缓存。因为这种拷贝是在内核完成的，sendfile()要比组合read()和write()以及打开关闭丢弃缓冲更加有效(更多有关于sendfile)

tcp_nopush 告诉nginx在一个数据包里发送所有头文件，而不一个接一个的发送

tcp_nodelay 告诉nginx不要缓存数据，而是一段一段的发送--当需要及时发送数据时，就应该给应用设置这个属性，这样发送一小块数据信息时就不能立即得到返回值。

```
1. access_log off;
2. error_log /var/log/nginx/error.log crit;
```

access_log设置nginx是否将存储访问日志。关闭这个选项可以让读取磁盘IO操作更快(aka,YOLO)

error_log 告诉nginx只能记录严重的错误

```
1. keepalive_timeout 10;
2.
3. client_header_timeout 10;
4. client_body_timeout 10;
5.
6. reset_timedout_connection on;
7. send_timeout 10;
```

keepalive_timeout 给客户端分配keep-alive链接超时时间。服务器将在这个超时时间过后关闭链接。我们将它设置低些可以让ngnix持续工作的时间更长。

client_header_timeout 和client_body_timeout 设置请求头和请求体(各自)的超时时间。我们也可以把这个设置低些。

reset_timeout_connection告诉nginx关闭不响应的客户端连接。这将会释放那个客户端所占有的内存空间。

send_timeout 指定客户端的响应超时时间。这个设置不会用于整个转发器，而是在两次客户端读取操作之间。如果在这段时间内，客户端没有读取任何数据，nginx就会关闭连接。

```
1. limit_conn_zone $binary_remote_addr
   zone=addr:5m;
2. limit_conn addr 100;
```

limit_conn_zone设置用于保存各种key（比如当前连接数）的共享内存的参数。5m就是5兆字节，这个值应该被设置的足够大以存储（32K*5）32byte状态或者（16K*5）64byte状态。

limit_conn为给定的key设置最大连接数。这里key是addr，我们设置的值是100，也就是说我们允许每一个IP地址最多同时打开有100个连接。

```
1. include /etc/nginx/mime.types;
2. default_type text/html;
3. charset UTF-8;
```

include只是一个在当前文件中包含另一个文件内容的指令。这里我们使用它来加载稍后会用到的一系列的MIME类型。

default_type设置文件使用的默认的MIME-type。

charset设置我们的头文件中的默认的字符集

以下两点对于性能的提升在伟大的WebMasters StackExchange中有解释。

```
1. gzip on;
2. gzip_disable “msie6”;
3.
4. # gzip_static on;
5. gzip_proxied any;
6. gzip_min_length 1000;
7. gzip_comp_level 4;
8.
9. gzip_types text/plain text/css
   application/json application/x-javascript text/
   xml application/xml application/xml+rss text/
   javascript;
```

gzip是告诉nginx采用gzip压缩的形式发送数据。这将会减少我们发送的数据量。

gzip_disable为指定的客户端禁用gzip功能。我们设置成IE6或者更低版本以使我们的方案能够广泛兼容。

gzip_static告诉nginx在压缩资源之前，先查找是否有预先gzip处理过的资源。这要求你预先压缩你的文件（在这个例子中被注释掉了），从而允许你使用最高压缩比，这样nginx就不用再压缩这些文件了（想要更详尽的gzip_static的信息，请点击这里）。

gzip_proxied允许或者禁止压缩基于请求和响应的响应流。我们设置为any，意味着将会压缩所有的请求。

gzip_min_length设置对数据启用压缩的最少字节数。如果一个请求小于1000字节，我们最好不要压缩它，因为压缩这些小的数据会降低处理此请求的所有进程的速度。

gzip_comp_level设置数据的压缩等级。这个等级可以是1-9之间的任意数值，9是最慢但是压缩比最大的。我们设置为4，这是一个比较折中的设置。

gzip_type设置需要压缩的数据格式。上面例子中已经有一些了，你也可以再添加更多的格式。

```
1. # cache informations about file descriptors,
   frequently accessed files
2. # can boost performance, but you need to test
   those values
3. open_file_cache max=100000
   inactive=20s;
4. open_file_cache_valid 30s;
5. open_file_cache_min_uses 2;
6. open_file_cache_errors on;
7.
8. ##
9. # Virtual Host Configs
10.     # aka our settings for specific servers
11.     ##
12.
13.     include /etc/nginx/conf.d/*.conf;
14.     include /etc/nginx/sites-enabled/*;
```

open_file_cache打开缓存的同时也指定了缓存最大数目，以及缓存的时间。我们可以设置一个相对高的最大时间，这样我们可以在它们不活动超过20秒后清除掉。

open_file_cache_valid 在open_file_cache中指定检测正确信息的间隔时间。

open_file_cache_min_uses 定义了open_file_cache中指令参数不活动时间期间里最小的文件数。

open_file_cache_errors指定了当搜索一个文件时是否缓存错误信息，也包括再次给配置中添加文件。我们也包括了服务器模块，这些是在不同文件中定义的。如果你的服务器模块不在这些位置，你就得修改这一行来指定正确的位置。

一个完整的配置

```
1. user www-data;
2. pid /var/run/nginx.pid;
3. worker_processes auto;
4. worker_rlimit_nofile 100000;
5.
6. events {
7.     worker_connections 2048;
8.     multi_accept on;
9.     use epoll;
10.     }
11.
12.     http {
13.         server_tokens off;
14.         sendfile on;
15.         tcp_nopush on;
```

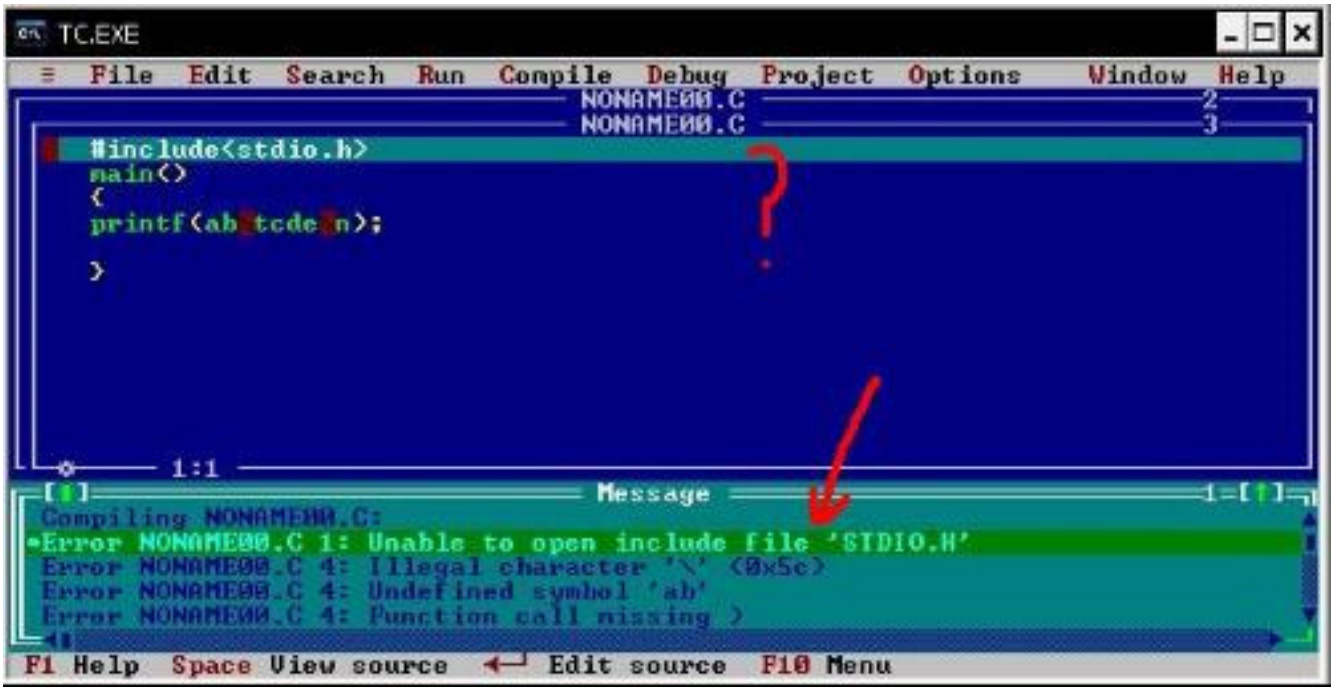
```
16.         tcp_nodelay on;
17.
18.         access_log off;
19.         error_log /var/log/nginx/error.log crit;
20.
21.         keepalive_timeout 10;
22.         client_header_timeout 10;
23.         client_body_timeout 10;
24.         reset_timedout_connection on;
25.         send_timeout 10;
26.
27.         limit_conn_zone $binary_remote_addr
    zone=addr:5m;
28.         limit_conn addr 100;
29.
30.         include /etc/nginx/mime.types;
31.         default_type text/html;
32.         charset UTF-8;
33.
34.         gzip on;
35.         gzip_disable “msie6”;
36.         gzip_proxied any;
37.         gzip_min_length 1000;
38.         gzip_comp_level 6;
39.         gzip_types text/plain text/css
    application/json application/x-javascript text/
    xml application/xml application/xml+rss text/
    javascript;
40.
41.         open_file_cache max=100000
    inactive=20s;
42.         open_file_cache_valid 30s;
43.         open_file_cache_min_uses 2;
44.         open_file_cache_errors on;
45.
46.         include /etc/nginx/conf.d/*.conf;
47.         include /etc/nginx/sites-enabled/*;
48.     }
```

编辑完配置后，确认重启nginx使设置生效。

```
sudo service nginx restart
```

后记

就这样！你的Web服务器现在已经就绪，之前困扰你的众多访问者的问题来吧。这并不是加速网站的唯一途径，很快我会写更多介绍其他加速网站方法的文章的。 ■



> “为什么我还要再用C？”-虽然我不同意他的说法，但至少他随口提到如果你“在一台拇指大小的电脑”上编程，或者为一门语言写引导程序，那么可以用C语言。要我说，写设备驱动，或者特定平台的内核，不管怎么说都可以使用C。

关于C语言，我喜欢和讨厌的十件事

前言：最近有个家伙抱怨道“为什么我还要再用C？”-虽然我不同意他的说法，但至少他随口提到如果你“在一台拇指大小的电脑”上编程，或者为一门语言写引导程序，那么可以用C语言。要我说，写设备驱动，或者特定平台的内核，不管怎么说都可以使用C。

几年之前，我用C语言写下了我的第一个网络程序，但我并不推荐这么做。现在，我只用P打头的，尤其是P-y打头的语言写网络程序（译者注：绕什么圈子，不就是Python嘛…）。但在当时，我刚从DOS和TSRs的世界中出来，在那儿用上10KB的RAM我都会觉得大得惊人。

现在我是一名Web开发者，但是仅限于晚上。白天我为嵌入式微处理器编写固件，因此，C依旧是我所选择的语言。我所说的微处理器是那种嵌入烤面包机，或者其他类似设备中的处理器，只有大概64KB的代码空间以及2KB的RAM。因此，可供选择的语言基本上就只有汇编和C了。（也可以是Forth，不过那是另外的故事。）

然后，我渐渐发现越是多用C，就越不觉得它讨厌了。因此我就想着要给这个世界最常用的系统级程序语言写一些颂词。

以下分别是关于C语言我喜欢五件事和讨厌的五

件事。请随意在底下的评论栏里加上你们自己喜欢或讨厌的事情。

1.K&R（喜欢）

Kernighan & Ritchie 写的《C程序设计语言》是关于C语言最好的书，而且我估计它也是关于编程的最好的书之一。简洁明了，都是有用并且重要的例子。这是一本非常好的书，同时也是一个非常好的参考。

甚至就连序言都非常好。在此引用一句，“C不是一门庞大的语言，因此不应该用一本厚重的书来诠释。”如果所有的编程教程都像这本书一样把长度限制到270页，它们会好很多。K&R的简洁明了、点到为止，很可能是C语言的成功所不可或缺的。

另一本给我喜爱的类似的编程教材是Leo Brodie所著的《Thinking Forth》。当然，肯定还有其他非常好的书，像是SICP之类的，只是我还没有读过罢了。

2.它十分简明（喜欢）

事实上，C语言作为一门简明语言是一个实实在在的福利。想要学习C，你只需了解它的类型，熟悉流程控制，处理好指针，然后你基本上就已经掌握它了。剩下的就仅仅是函数了。事实上，K&R利用这个低级的命令式语言，仅花费11行就实现了qsort()，不得不说这是对C语言简明性有力的证明。

3.IOCCC（喜欢）

你或许会觉得我疯了，不过如果你足够上进，International Obfuscated C Code Contest可能那儿是关于计算机科学最好的老师之一。算我开的一个小玩笑，不过我的确认为众多黑客都在不停挑战，并且创造了很多值得一谈的功绩。

其中让我确实学到很多的就是OTCC，Fabrice Bellard所写的“混淆的小型C编译器”。从中我学到了关于编译器设计的知识，主要是C语言编译器不必是340万行代码的庞然大物。同时，我也从Let’s Build a Compiler中获益，并静下来写了一个迷你的由C到Forth的编译器。

4.变量的定义与使用形式相似（喜欢）

这一点对记住如何定义十分复杂的事物非常有用，举个例子，一个指向包含十个整形的数组的指针应该是int *api[10]还是int (*pai)[10]呢？像你使用它的方式一样定义它即可，只需要记住[]操作符的优先级高于*（很自然就可以记住），然后你就明白那个括号是需要的了。(译者注：前者是指针数组，包含十个指向整形的指针。)

5.它编译出的“hello, world”体积很小（喜欢）

尤其是对嵌入式编程，这一点简直棒极了。C语言之上没有一个体积庞大的运行时，在很多嵌入式处理器上，一个什么都不做的程序一般只会编译出3到4个byte。一个完整的“hello, world”程序，甚至是在Windows XP下，都只会编译出1.5KB大小（使用Tiny C Compiler，它非常合适与做小型可运行程序）。

我认为，如果像Python一样的其他语言能够在这一点上赶上C，甚至是C的一部分，他们在嵌入式的世界中就会更加出彩。

6. 全局变量默认是外部的（讨厌）

你会说“用全局变量可不是个好习惯！”。但

在嵌入式系统中不同。举个例子，你有一个名为timer.c的文件，其中有个全局变量int counter，在另一个文件state_machine.c中，有另一个counter。如果你碰巧忘记了在它们之前加上’static’，它们就是同一个变量，你根本察觉不到，没有Warning，没有任何提示……

这种行为看起来十分奇怪，尤其是当关键字extern就在手边的时候。不过当你熟悉static的两种不同的意义后，就可以轻易避免这种情况了。不过这依然十分令人讨厌。

7. static的两种不同的意义（讨厌）

有人能解释一下为什么static在函数体中和函数体外有着两种完全不同的意义吗？在函数体中，他表示“静态”——“在函数调用过程中保持这个变量唯一”。但是在函数体外，它的意义完全改变，成了“该变量为该文件私有的”。为什么后者不用private或者intern呢？

8. & 优先级低于 ==（讨厌）

在嵌入式编程中，我们总是喜欢用if((x&MASK) == 0)这样的语句。但你可能常忘记写里面那对括号，因为感觉上，&的优先级应该比==高。但是事实并非如此，因此必须使用这对多出来的括号。

不过，这个情况有个不错的历史原因。C语言诞生自B语言，而在B语言中只有&而没有&&运算符。当Ritchie引入&&运算符时，他们希望原有的B语言端的代码能够正常运行，因此使&的优先级低于==。

9. 宏的功能并没有那么强（讨厌）

虽然递归的#include是非常棒的点子，但是，要怎么做才能不诉诸一些费脑子的方法，轻易地做预处理循环呢？同样的，有些我常遇到的情况，比如怎么才能给程序int和string两种格式的版本号，而同时只需要修改一个变量呢？

```
1. #define VERSION_INT 209
2. #define VERSION_STR  “2.09”
```

用上面的代码，你更新版本号的时候总是需要修改两个地方。而且，特殊的#和##并不能帮上什么忙。我找到的唯一的解决则涉及了一些运行时修改。

10. 它不支持反射（讨厌）

好吧，可能这只是重申了一下第9点——如果宏系统再稍微强大一点，就不需要反射机制了。说不定我还会滥用它。不过我真正想说的是，用C语言，你不能写出生成代码的代码。

为什么不用C语言本身来写预处理器呢？这会给循环展开，更强大的宏机制，甚至更多IOCCC的怪点子提供无穷无尽的可能性。:-)

我认为，C语言之父能够坦然承认C的不足之处是非常可贵的。就像Dennis Ritchie说的一样：

“C语言行为古怪，瑕疵遍布，但却是一个巨大的成功。”

更多关于这点的信息，去读读他的论文 The Development of the C language 吧 —— 那真是一篇值得一读的文章。

总而言之，在自己的优势上，C卓尔不群。■

程序员，你调试过的最难的Bug是什么？

编者按

调试 Bug?每个程序员工作中必须品。在 Quora 上有一个和 Bug 相关的热门问答帖：《你调试过的最难 Bug 是？》。在众多回复中，Dave Baggett 的经历最让人惊叹，得到了 2400 多个顶。

回想起这个bug，仍然让我有些痛苦。作为一个程序员，在发现bug时，你学会了首先在自己代码中找问题，或许在测试一万次之后，你会把问题归咎于编译器。只有在这所有的都不起作用之后，你才会把问题归咎于硬件。

这是我遭遇一个硬件bug的故事。

抛开别的不说，我曾为《Crash Bandicoot》写存储卡（读写）代码。对于一个自大的游戏程序员，这就像是在公园里散步一样轻松愉快，我认为只要几天就写完了。我中止调试六个礼拜。在此期间我做一些其他的事情，但我一直回来处理这个bug——几天内每天几个小时。这个bug实在烦人。

这个bug的症状是，当你需要保存你的进度时，代码会访问存储卡，而大部分情况下没有什么问题…但是偶尔读写会超时…没有任何明显的原因。一个短小的写入经常毁掉存储卡。玩家要保存进度，我们不仅不保存，还擦除他们存储卡上的全部东西。天哪。

过了一段时间，我们在Sony的制作人Connie Booth慌了。我们显然不能带着这个bug发布游戏，而六个星期之后我对于问题出在哪一点线索都没有。通过Connie我们向其他 PS1 开发者求助：有没有人出现过像我们这样的情况？没有。绝对没有任何人在存储卡系统上出现任何问题。

在你绞尽脑汁之后，你能做的唯一一个调试方法就是分而治之：一点点去除程序中的代码，直到留下的代码很少但你仍然出问题。像木雕一样去除没有问题的代码，留下的就是你的bug所在。

在这样的背景下挑战在于，视频游戏是很难去除某一部分的。在你删除模拟重力或者显示字符的代码后，如何运行游戏？

你必须做的是用一个假装做真正的事情，但实际上只是做很简单的不会出现bug事情的东西来替换掉整个模块。你必须写新的支撑代码来让这些玩意正常工作。这是一个缓慢而痛苦的过程。

长话短说：我做完了。我移除了大片大片的代码，相当多，只留下了初始化代码——就是准备游戏运行系统，初始化底层硬件等等。当然，我不能显示加载/保存菜单，因为我截除了所有的图像代码。但是我能够假装用户使用（不可见的）加载/保存屏幕并且请求保存，然后写入卡中。

我最终以一个带有这个bug的很少量的代码结束——但问题仍然随机出现！在大多数情况下没啥问题，但是偶尔会失效。基本上所有的Crash的实际代码都被移除了，但还是这样。这实在是莫名其妙：留下来的代码基本上都没做什么事。

在那时——估计是凌晨3点——一个想法蹦了出来。读写（I/O）涉及精确定时。无论是硬盘、存储卡、蓝牙发送器——随便啥——做读写的底层代码都是根据时钟来的。

时钟让不直接连接到CPU的硬件设备和cpu运行的代码同步。时钟决定了波特率——数据从一头传到另一头的速率。如果计时有什么问题，硬件或者软件或者两者都会乱七八糟的。这真的，真的很糟糕，并且通常导致数据损坏。

如果我们的初始化代码以某种方式弄乱了计时会怎么样？我又看了一遍测试程序中和计时有关的代码，并注意到我们将PS1上的可编程计时器设置到了 1kHz（1000跳每秒）。这是比较快了，当PS1启动的时候，默认状态大概是100Hz。因此，大多数游戏将他们的计时器设置为100Hz。

这个游戏的带头（和除我外的唯一）开发者Andy，将计时器设置为1kHz，使得Crash的动作计算更加准确。Andy喜欢矫枉过正，如果我们要模拟重力，我们应该尽可能的提高精度！

然而如果提高计时器频率莫名其妙的干扰了整个程序的计时，故而将这个计时器设置到存储卡的波特率上会怎样呢？

我将计时器代码注释掉。然后我就无法复原这个bug了。但是这并不表示bug被修复了，这个问题是随机发生的。万一我只是运气好呢？

几天过去了，我还是在玩我的测试程序。Bug没有再出现。我回到全部的Crash代码中，修改了加载/保存代码，在访问存储卡之前将可编程计时器重置为默认设置（100Hz），之后设置回1kHz。从此之后没有发现问题再次出现。

但是…为什么？

我重新回到测试程序上，试着检测当计时器设置为1kHz时出现的那些错误的模式。终于，我注意到这些错误出现在使用PS1手柄的人身上。因为我自己 很少这样做，所以我没有注意到（为啥我要在测试加载/保存代码的时候用手柄）。但是有一天我们的美工等我去完成测试（我确定那时候我在爆粗口），而他紧张 的摆弄着手柄。卡损坏了。“等下，怎么回事？喂，再来一次！”

一旦我发现了这两件事是联系着的，就很容易重现bug：开始写入存储卡，动一下手柄，存储卡损坏。在我看来完全是硬件bug。

我去找Connie告诉他我的发现。她转述给设计过PS1的硬件工程师。她被告知：“不可能，这不可能是硬件问题。”我跟她说问一下我能不能直接和他说。

那个工程师给我打电话了，他用着他的烂英语，我用着我更烂的日语，我们争论一会。我最后说：“我给你一个30行的测试程序，让你在动手柄的时候能够 出现这问题。”他答应了。他向我保证，这是浪费时间，而他正在一个新项目上很忙，但因为我们是Sony很重要的开发者，他会试的。

第二天晚上（我们在洛杉矶，而他在东京，所以对于我来说是晚上而他是到了第二天），他给我打电话，不好意思的向我道歉。这是个硬件问题。

我还是没有完全搞清楚问题到底在哪，但是我的印象中，从Sony总部的反馈听到的是，如果将可编程计时器设置到足够高的时钟频率，会影响到主板上时 钟晶振附近的一些东西。这些东西之一就是存储卡的波特率控制器，同时也设置手柄的波特率。我不是搞硬件的，所以对于细节我相当模糊。

但是主旨是主板上两个独立部分的串扰，以及手柄接口和存储卡接口数据发送的结合在1kHz的时钟频率下会导致丢位，从而数据丢失，以致卡损坏。

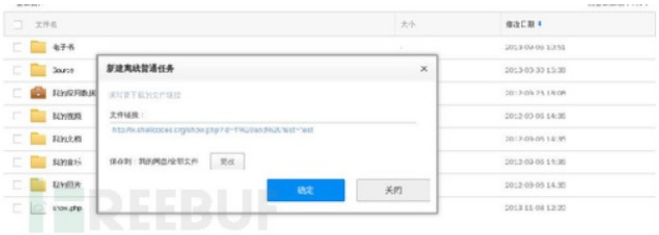
这是我全部编程生涯中，唯一一次因为量子力学debug的问题。■

求：http://lx.shellcodes.org/show.php?id=4 and 1=1

订阅这条带payload的地址后，我从博客的日志中就发现了攻击请求：

这个IP其实就是鲜果订阅的IP。

百度网盘也可以，新建个下载任务：



接着看日志：

像百度网盘这些提供了API服务的，是不是写点脚本批量攻击呢;还有一些服务可以看到请求后返回的HTTP状态的，是不是更精准知道攻击结果呢。。理论上可以绕过一些IP白名单，剩下的看官们自己发挥吧，我继续工作了。■

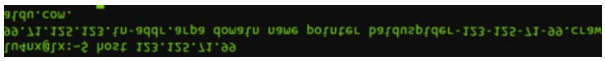
技 巧

移花接木，如何利用第三方服务发动WEB攻击

继《走近科学：如何利用Google机器人进行SQL攻击》后，咱们来讨论讨论如何利用三方服务进行移花接木。

在很久之前，我们就在加速乐的离线日志中发现了来自谷歌爬虫攻击，事实上，不止谷歌爬虫，包括百度之类的也在内，如下图，是我们一个内部平台找到的一条来自百度爬虫的注入：

验证下这个IP：123.125.71.99：



除了百度、谷歌，还有其他。利用其他三方网站的服务去请求payload实际上是很容易的。例如，我还从加速乐日志里发现来自google reader等等。

来点实际的。比如，我可以用鲜果阅读的订阅去向我博客提交注入请

极客无极限：一行HTML5代码引发的创意大爆炸

编者按

一行HTML5代码能做什么？国外开发者Jose Jesus Perez Aguinaga写了一行HTML5代码的文本编辑器。

一行HTML5代码能做什么？国外开发者Jose Jesus Perez Aguinaga写了一行HTML5代码的文本编辑器。这件事在分享到Code Wall、Hacker News之后，引起了众多开发者的注意，纷纷发表了自己的创意。 这是最初的HTML5代码，它可以运行在最新的Chrome和Firefox中，只需在浏览器地址栏输入如下代码：

```
1. data:text/html, <  html   contenteditable>
```

但是功能十分有限，甚至没有保存功能，样式也非常简陋。 于是，网友Montas修改了他的代码，使用textarea标签代替html标签，可以添加自己喜欢的样式：

```
1. data:text/html, <textarea style=”font-size:
   1.5em; width: 100%; height: 100%; border:
   none; outline: none”  autofocus />
```

网友jecxjo希望能有存储功能：

```
1. data:text/html,<button
   onClick=”SaveTextArea()”>Save</button>
   <script language=”javascript”  type=”text/
   javascript”> function SaveTextArea() {
   window.location = “data:application/octet-
   stream,” + escape(txtBody.value); } </script>
   <textarea id=”txtBody”  style=”font-size:
   1.5em; width: 100%; height: 100%; boarder:
   none; outline: none”  autofocus> </textarea>
```

但上面的代码是以文件形式存储，samsonjs觉得不够方便，而且需要点击按钮，于是添加了自动保存功能：

```
1. data:text/html,<html lang=”en”><head><style>
   html,body { height: 100% } #note { width:
   100%; height: 100% } </style> <script> var
   note, html, timeout; window.
   addEventListener(‘load’, function() { note =
   document.getElementByIld(‘note’); html =
   document.getElementsByTagName(‘html’)
   [0]; html.addEventListener(‘keyup’,
   function(ev) { if (timeout)
   clearTimeout(timeout); timeout =
   setTimeout(saveNote, 100); }); restoreNote();
   note.focus(); }); function saveNote() {
   localStorage.note = note.innerText; timeout =
   null; } function restoreNote() { note.innerText =
   localStorage.note ||  ‘’ ; } </script> </
   head><body><h1>Notepad (type below, notes
   persist)</h1> <p id=”note”
   contenteditable=”” ></p> </body></html>
```

现在可是云时代！仅仅这样怎能让开发者止步？minikomi使用了第三方API打造了一个在线编辑器：

```
1. data:text/html,
2. <style type=”text/css”>
```

```
3. #e {
4. position:absolute;
5. top:0;
6. right:0;
7. bottom:0;
8. left:0;
9. font-size:16px;
10.     }
11.     </style>
12.     <div id=”e”></div>
13.     <script src=”http://d1n0x3qji82z53.
    cloudfront.net/src-min-noconflict/ace.js”></
    script>
14.     <script src=”http://code.jquery.com/
    jquery-1.9.0.min.js”></script>
15.     <script>
16.     var myKey=”SecretKeyz”;
17.     $(document).ready(function(){
18.     var e;
19.     var url = “http://api.openkeyval.
    org/”+myKey;
20.     $.ajax({
21.     url: url,
22.     dataType: “jsonp”,
23.     success: function(data){
24.     e = ace.edit(“e”);
25.     e.setTheme(“ace/theme/tomorrow_
    night_eighties”);
26.     e.getSession().setMode(“ace/mode/
    markdown”);
27.     e.setValue(data);
28.     }
29.     });
30.
31.     $(“#e”).on(“keydown”, function (b) {
32.     if (b.ctrlKey && 83 == b.which) {
33.     b.preventDefault();
34.     var data =
    myKey+”=”+encodeURIComponent(e.
    getValue());
35.     $.ajax({
36.     data: data,
37.     url: “http://api.openkeyval.org/store/”,
38.     dataType: “jsonp”,
39.     success: function(data){
40.     alert(“Saved.”);
41.     }
42.     });
43.     }
44.     });
45.     });
46.     </script>
```

没有代码高亮功能的编辑器终究不适合程序员，Rails开发者jakeonrails又定制了Ruby代码高亮功能：

```
1. data:text/html, <style type=”text/css”>#e{po
    sition:absolute;top:0;right:0;bottom:0;left:0;}</
    style><div id=”e”></div><script src=”http://
    d1n0x3qji82z53.cloudfront.net/src-min-
    noconflict/ace.js” type=”text/javascript”
```

charset=”utf-8”></script><script>var e=ace.edit(“e”);e.setTheme(“ace/theme/monokai”);e.getSession().setMode(“ace/mode/ruby”);</script>

效果如下图：

```ruby
# The Greeter class
class Greeter
  def initialize(name)
    @name = name.capitalize
  end

  def salute
    puts "Hello #{@name}!"
  end
end

# Create a new object
g = Greeter.new("world")

# Output "Hello World!"
g.salute
```

实际上，如果minikomi的代码已经支持多种语言高亮，如Python，只需要把“markdown”换成“python”，效果如下：

```python
def row(x):
    return ' '.join(map(str, reduce(lambda x,
        y: map(sum,zip([0]+x,x+[0])),range(x),[1])))
def pascal(x):
    return '\n'.join(row(i).center(len(row(x-1))) for i in range(x))
print pascal(10)
```

你以为到此为止了？中国开发者assassindesign觉得只是文本编辑器就太无聊了，又提供了涂鸦版本：

1. data:text/html, <body><canvas id=”dyDraw”>你的浏览器不支持HTML5 Canvas</canvas></body><script>function $(id){return document.getElementById(id);} $(‘dyDraw’).width=document.body.clientWidth;$(‘dyDraw’).height=document.body.clientHeight;if(window.addEventListener){window.addEventListener(‘load’,function(){var canvas,canvastext;var hua=false;function dyDrawing(){canvas=$(‘dyDraw’);canvastext=canvas.getContext(‘2d’);canvas.addEventListener(‘mousedown’,canvasMouse,false);canvas.addEventListener(‘mousemove’,canvasMouse,false);window.addEventListener(‘mouseup’,canvasMouse,false);} function canvasMouse(dy){var x,y;if(dy.layerX||dy.layerX==0){x=dy.layerX;y=dy.layerY;}else if(dy.offsetX||dy.offsetX==0){x=dy.offsetX;y=dy.offsetY;}x-=dyDraw.offsetLeft;y-=dyDraw.offsetTop;if(dy.type==’mousedown’){hua=false;canvastext.beginPath();canvastext.moveTo(x,y);hua=true;}else if(dy.type==’mousemove’){if(hua){canvastext.strokeStyle=”rgb(255,0,0)”;canvastext.lineWidth=9;canvastext.lineTo(x,y);canvastext.stroke();}}else if(dy.type==’mouseup’){hua=false;}} dyDrawing();},false);}</script>

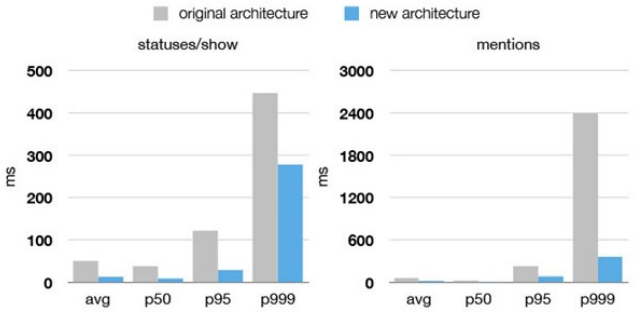
■

性能大幅度提升的Twitter新系统架构

□ IT经理网/关志刚

8月3日《天空之城》在日本的热播创下每秒新增143119条推文的Twitter峰值记录，是Twitter平均每秒发推数（TPS）5700条的25倍。

值得注意的是，在这次毫无征兆的“洪峰”到来时，Twitter全新的系统平台并没有被潮水般涌来的推文堵塞而产生任何延迟甚至宕机。



Twitter旧架构与新架构的性能对比

仅仅三年前，在2010年世界杯上，一个点球和一张红牌产生的“推文风暴”都可能导致Twitter服务暂时失去响应，号称地球脉搏的Twitter经常“心肌梗塞”。过去三年Twitter的工程师们夜以继日的工作，试图用“缝缝补补”的方式完善Twitter系统，但最终随着Twitter的快速发展，这些方法的收效转瞬即逝。

最终，Twitter痛下决心重新架构为人诟病的IT系统，新平台上线运行后在性能和可靠性上都取得的翻天覆地的进步。无论是《天空之城》热播还是超级碗决赛都没能卡住Twitter，而且新的架构也为Twitter推出多媒体推文卡片，跨设备消息同步等新功能的推出提供了有力的支撑。

最近，Twitter平台工程副总裁Raffi Krikorian（@raffi）在Twitter官方博客撰文分享了Twitter新架构的方法和经验，摘要如下：

重新架构的缘由与问题症结

2010年世界杯多次卡壳后，我们重新审视了系统，有以下几点发现：

我们运行着全球最大的Ruby on Rails应用，200名工程师负责开发运维这个系统，但随着用户规模和服务数量的快速增长，系统所有的数据库管理、Memcache链接以及公共API的代码属于同一个代码库。这给工程师的学习、管理和并行开发都带来巨大困难。

我们的MySQL存储系统已经遇到性能瓶颈。整个数据库中到处都是读写热点。

通过添置硬件已经无法解决根本的系统问题——我们的前端Ruby服务器每秒处理交易的数量大大低于我们的预期，也与其硬件性能不成比例。

从软件的角度看，我们陷入了“优化的陷阱”。我们是在牺牲代码库的可读性和灵活性来换取性能和效率。

重新检视系统，并设定三大目标/挑战

一、新架构必须在性能、效率和可靠性上表现优异，减少延迟大幅提升客户体验；同时将服务器数量减少到原来的十分之一；新系统能够隔离硬件问题防止其演变为大规模宕机。

二、解决单一代码库的种种弊端，尝试松耦合的面向服务模型。我们的目标是鼓励封装与模块化的最佳实践，但这次是在系统层面，而不是类库、模块和数据包的层面。

三、最重要的是能够支持新功能的快速发布。我们希望能够由一些充分授权的小团队能做出自主决策，并独立发布一些用户功能。

我们在动手前部分开发了一些概念验证模型，最终我们确定了重建的原则、工具和架构。

系统重建的关键措施

一、前端服务：用JVM取代Ruby VM。通过重写代码库将Ruby VM服务移植到JVM，性能提高了10倍，如今性能达到 10-20k请求/秒/主机。

二、编程模型：按服务类型对系统进行结构，建立一个统一的客户端服务器库并与负载均衡、故障转移策略等绑定，从而让工程师们能更加专注于应用和服务界面。

三、采用SOA面向服务架构，使并行开发成为可能。

四、推文的分布式存储。即使将整块单一应用分解成不同的“服务”，存储依然是个巨大的瓶颈。过去Twitter采用的单一MySQL主数据库只能线性写入推文，Twitter决定在推文的存储上采用全新的分区策略，用Gizzard框架创建容错的分片分布式数据库存储推文，但这样一来就没有办法使用MySQL的唯一ID生成功能。Twitter用Snowflake解决了这个问题。

五、监测与统计。将单一应用转化为复杂的SOA应用后，需要购买匹配的工具才能够驾驭。Twitter的服务推出速度很快，同时还需要实现数据化的决策支持，Twitter的Runtime系统团队为工程师开发了两个工具Viz和Zipkin。■

专题推荐

浅谈当下网页设计趋势

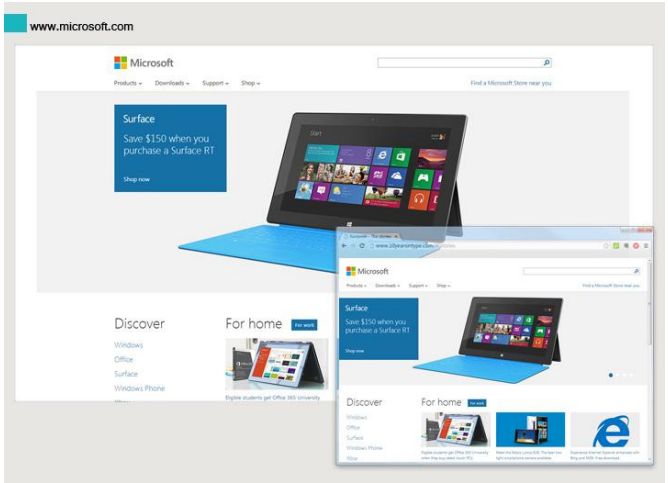
在我们适应着现代设计潮流的同时，不妨也来看看现阶段网页设计大致的趋势和风格吧。我不敢大言不惭的说这就是当下网页设计的趋势，这只 是本人对当下网页设计做出的一些小总结。希望这样的归类总结能给你带来更多的思路和想法。

技术的革新带动了设计行业的的迅猛发展，这使得设计师和开发者有了更广阔的的探索天地。而网页设计也越发不再那么循规蹈矩，许多团队和公司都做了很 多思考和创意。所以在我们适应着现代设计潮流的同时，不妨也来看看现阶段网页设计大致的趋势和风格吧。我不敢大言不惭的说这就是当下网页设计的趋势，这只 是本人对当下网页设计做出的一些小总结。希望这样的归类总结能给你带来更多的思路和想法。

1.响应式网页设计（Responsive Web Design）

现在越来越多用户都拥有多种终端：台式机，笔记本，平板电脑，手机，能够适应不同尺寸显示屏的网页是现在的潮流，甚至是未来很长一段时间的设计趋 势。那么响应式网页设计就是来解决这个问题的。这种特别的开发方式能保证网页适应不同的分辨率，让网页要素重组，使其无论在垂直的平板电脑还是智能手机上，都能达到最好的视觉效果。

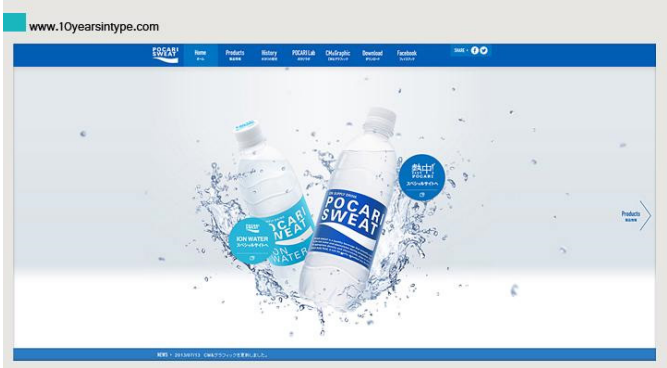
除了多终端的多样化，我们还可以看到我们的电脑屏幕，手机屏幕都在不断变大，而在对未来生活的预测、概念设计里，“屏幕”这个产物更是被运用到多种 新平台上。例如微软发布的“未来生活概念视频”里，厨房、室内墙壁、办公室玻璃墙面都成为了交互平台。所以我们可以发现，响应式网页设计所具备的良好的适 应性和可塑性，在未来的网页设计里将占有举足轻重的位置。



2.全屏网页设计（Full Screen Design）

所谓设计不分家，近年来平面设计里“纯净”“留白”等概念也被互联网设计吸取，为了更简单明了的突出主体，提供更舒适的感官感受，很多网站开始采用 全屏网页设计，利用精心挑选设计

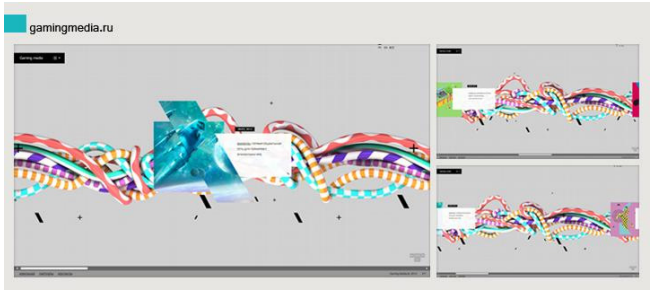
的漂亮背景，加上合理的页面布局，视觉冲击力大可很好的吸引观者注意。通常页面内的文字内容不会特别多（所出现的少量文字 加上精美的排版将会变得更加吸引人），主要以图片展示为主。这个样子的网站多用于摄影团队或个人作品集展示会比较常见。虽然简单养眼，但是承载信息有限， 公司部门的主页很少见这样的设计。



3.视差滚动设计（Parallax Design）

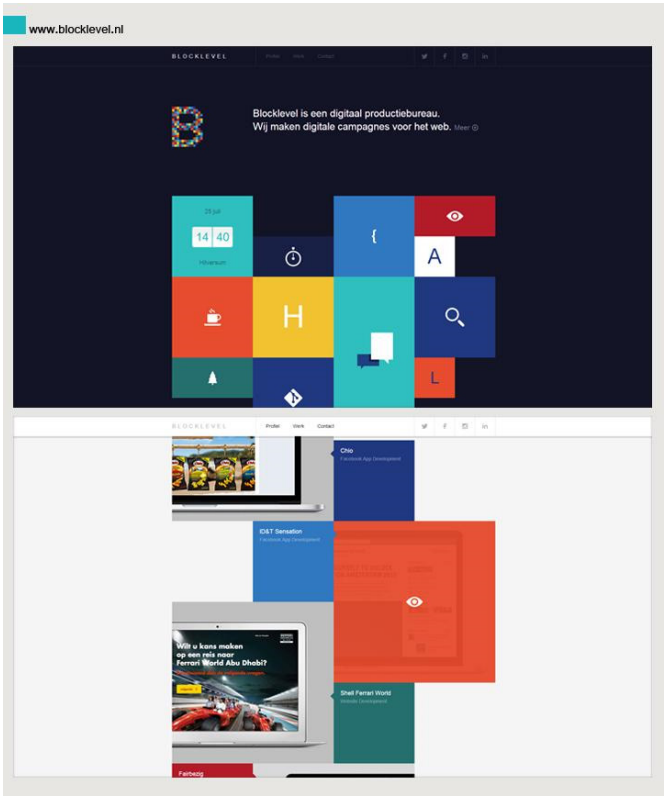
视差设计可以说是近年来网页设计中的一大突破，也备受推崇。视差滚动是让多层背景以不同速度滚动，以形成一种3D立体的运动效果，给观者带来一种独特的视觉感受。

除此以外，鼠标滚轮的流畅体验，让用户在观看此类网站时有一种控制感，简单来说这是有响应的交互体验。就好像童年看到走马灯，转动它你就能看到人物 动起来，还能欣赏故事。视差滚动设计的趣味也在于此。所以无论是网站还是电商商品宣传页都经常采用视差设计，吸引眼球也很受用户喜爱。



4.扁平化设计（Flat Design）

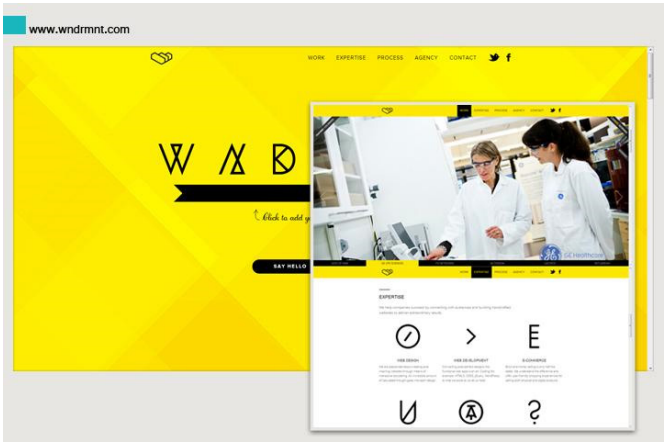
扁平化设计可以说是去繁从简的设计美学。去掉所有装饰性的设计，可以说是对之前所推崇的拟物化设计的颠覆。我们不能妄加评论说这是好还是不好，只能 说它提供了一种新的设计思维。扁平化设计是否会成为将来的趋势我们也无法回答，尽管褒贬不一，备受争议，但是就现在来说它是当下的一种潮流。

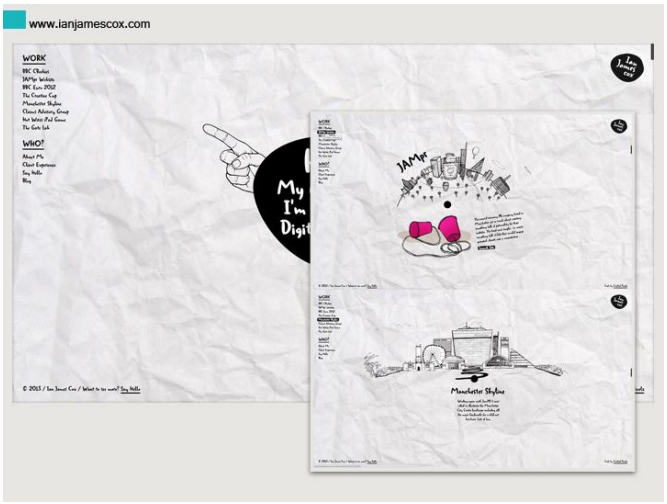


5.滚动侦测网页设计（Scrollspy）

利用CSS的实现将导航栏固定在网页顶部（大多数是顶部，当然也有侧面或底部），并将版面内容按照导航顺序垂直或横向排布，使得用户点击相应导航 tab时页面自动滑到相应页面，而若点击内容，导航也将随之改变。这样的网页设计页面基本不会跳转，每一个tab所指向的页面内容也基本一屏显示完整，所 以在页面呈现的内容上会有所局限。为不影响布局一般也会伴随自适应。

滚动侦测式的网页会给设计师带来了很大挑战——要在有限空间内保证内容呈现的完整性，故设计师会在版面上下足功夫。而这类网站结构和视差设计有异曲同工之处，所以我们发现很多网站会结合两者，给观者带来不一样的视觉感受和用户体验。





6.无限滚动模式（瀑布流）

有一些网站内容很多，但他们并没有简单分页，而是采用的是一种垂直瀑布流的方式布局。将那些内容垂直排布，当用户纵向滚动时，内容会不断更新好像永 无止境。这样的瀑布流很早之前就开始流行，最早采用该布局的是pinterest。这样的滚动页面就大大减少了分页的数量，个人认为对于这类信息量大，每 日更新数据快的网站是比较不错的方案。

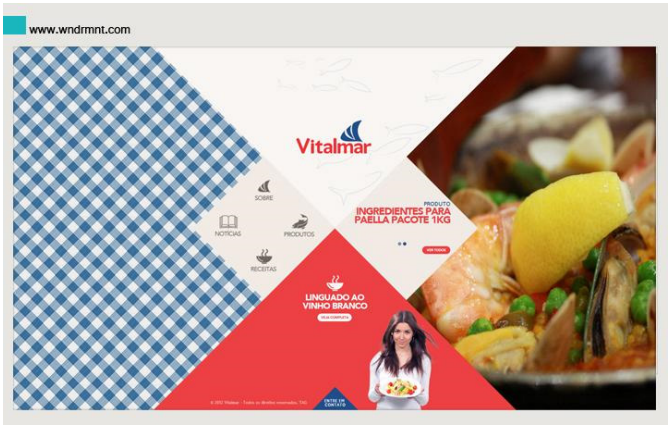
7.网页的风格化设计

现在的网页早已不再像过去受诸多条件和技术限制了。其呈现方式颇为丰富。风格从清新到复古，插画手绘到拟真设计，无奇不有。无论是版面版式，还是设计元素，用标新立异这个词形容绝不

为过。根据自己撇到的冰山一角，提一下对我感触最深的变化：

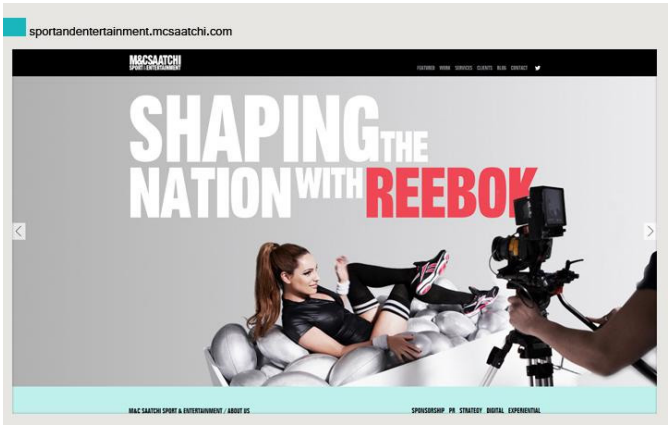
（1）平面设计感的加强

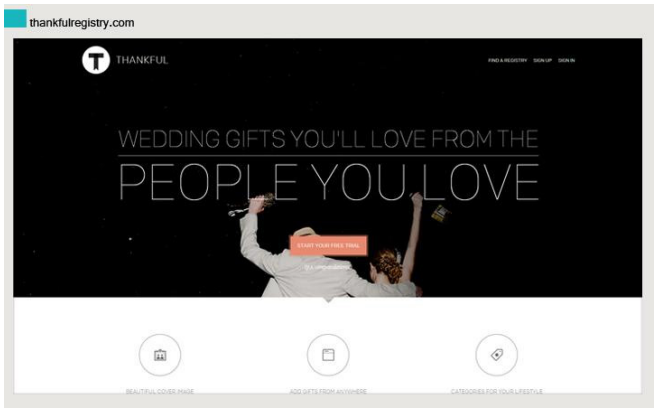
网页设计随着设备和技术的革新，早已突破了过去单一框架的限制，变得更加灵活。所以就页面风格更多地开始向平面设计靠近，许多页面设计得极赋海报和杂志的版式感。时尚而富有冲击力。



（2）注重字体设计

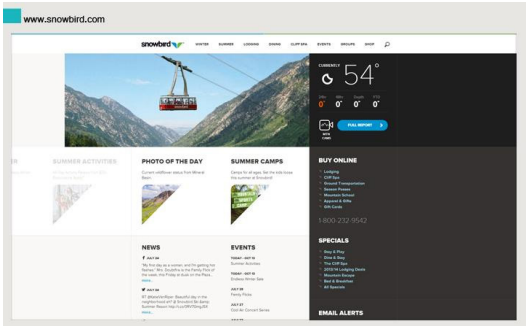
近年来很多设计师将字体设计也融入了网页设计中，并作为设计的一个重要元素提升整个网页品味。通过使用CSS3设计师可以拥有许多自定义的字体，这给网页的视觉设计也增加了一个重要的设计思路。





（3）丰富灵活的动画

Html5和flash的广泛应用，让网页的交互动画变得更加生动有趣。



通过观察这些趋势如何影响现代网站设计，或许可以为网页设计师带去指引，发散出新想法。

虽然设计师和开发者都需要和市场接轨并跟上潮流的脚步，但是所谓的潮流是当下的，未来确是未知的。我们的确需要保证自己不被行业趋势甩到队尾，但更重要的是在浪潮中适应和学习。■

封面设计：@51CTO小林

《开发月刊》电子杂志

51CTO开发频道

数据